TUGAS AKHIR – K141502

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DENGAN KENDALI *VOICE RECOGNITION*

WILDAN LUTFI SYARIATI FIRDAUS SYAWAL
NRP 5114100080

Dosen Pembimbing
Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Imam Kuswardayan, S.Kom., M.T.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – K141502

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DENGAN KENDALI *VOICE RECOGNITION*

WILDAN LUTFI SYARIATI FIRDAUS SYAWAL
NRP 5114100080

Dosen Pembimbing
Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Imam Kuswardayan, S.Kom., M.T.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – K141502

VIRTUAL REALITY GAME MAGUS USING OCULUS RIFT WITH VOICE RECOGNITION CONTROL

WILDAN LUTFI SYARIATI FIRDAUS SYAWAL
NRP 5114100080

Advisor
Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Imam Kuswardayan, S.Kom., M.T.

INFORMATICS DEPARTMENT
Faculty of Information and Communication Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DENGAN KENDALI *VOICE RECOGNITION*

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Interaksi, Grafika dan Seni
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
WILDAN LUTFI SYARIATI FIRDAUS SYAWAL
NRP: 5114100080

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Dr. Eng. Darlis Herumurti, S.Kom., M.Kom. .....
NIP: 19771217 200312 1 001 (pembimbing 1)

Imam Kuswardayan, S.Kom., M.T. .....
NIP: 19761215 200312 1 001 (pembimbing 2)

SURABAYA
MEI 2018

[Halaman ini sengaja dikosongkan]

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DENGAN KENDALI *VOICE RECOGNITION*

Nama Mahasiswa	: Wildan Lutfi Syariati Firdaus Syawal
NRP	: 5114100080
Departemen	: Informatika FTIK-ITS
Dosen Pembimbing 1	: Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Dosen Pembimbing 2	: Imam Kuswardayan, S.Kom., M.T.

ABSTRAK

Perkembangan teknologi dengan menggunakan realitas virtual sudah semakin maju. Tidak hanya untuk membantu manusia dalam mensimulasikan aktifitas di dunia nyata, juga menjadi sarana untuk bermain permainan yang terasa nyata. Untuk merasakan teknologi realitas virtual ada banyak macam alat yang dapat digunakan, salah satunya adalah Oculus Rift. Alat ini berfungsi untuk menampilkan dunia maya kepada pengguna.

Ide yang digunakan dalam tugas akhir ini adalah membangun sebuah permainan First Person Shooter (FPS) yang memiliki control Leap Motion dan Voice Recognition. Permainan ini menjadikan tangan dan suara sebagai sumber masukan dalam melakukan interaksi. Permainan ini dibangun menggunakan Unity dan Leap Motion SDK. Tujuan dari permainan ini adalah untuk meningkatkan adrenalin pemain untuk mengalahkan musuh yang ada sebelum musuh mengalahkannya lebih dulu.

Hasil dari pengujian ini akan berupa sebuah permainan yang dapat berjalan di perangkat personal computer high-end. Permainan ini dibangun dengan Unity Versi 2017.3.1.1f1 dengan bahasa pemrograman C#, Library Windows.Speech, dan Leap Motion SDK. Proses pembuatan asset permainan sebagian besar

mengambil dari asset store dan sebagian dibuat menggunakan tools Blender, Corel Draw X7, dan Adobe After Effects 2017cc. Dengan pengujian beta dan juga kuisioner dapat disimpulkan permainan telah mengimplementasikan perancangan dengan baik.

Kata kunci: Oculus Rift, Realitas Virtual, Voice Recognition

VIRTUAL REALITY GAME MAGUS USING OCULUS RIFT WITH VOICE RECOGNITION CONTROL

Name : Wildan Lutfi Syariati Firdaus Syawal
NRP : 5114100080
Department : Informatics FTIK-ITS
Supervisor I : Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Supervisor II : Imam Kuswardayan, S.Kom., M.T.

ABSTRACT

The development of technology by using virtual reality is increasingly advanced. Not only to help humans in simulating activities in the real world, it also becomes a means to play a game that feels real. To experience virtual reality technology there are many different tools that can be used, one of them is Oculus Rift. This tool serves to show virtual world to users.

The idea used in this final project is to build a First Person Shooter (FPS) game that has Leap Motion and Voice Recognition control. This game makes hands and sounds as a source of input in the interaction. The game is built using Unity and Leap Motion SDK. The purpose of the game is to increase the adrenaline of players to defeat the enemies before the enemy defeats them first.

The result of this testing will be a game that can run on high-end personal computer devices. The game is built with Unity Version 2017.3.1.1f1 with c# programming language, Windows.Speech Library, and Leap Motion SDK. The process of making asset games mostly take from the asset store and partly made using tools Blender, Corel Draw X7, and Adobe After Effects 2017cc. With beta testing and also the questionnaire can be concluded the game has implemented the design properly.

Keywords: Oculus Rift, Realitas Virtual, Voice Recognition

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah Subhanahu Wa Ta'ala karena atas karunia dan rahmat-Nya penulis dapat menyelesaikan tugas akhir yang berjudul:

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DENGAN KENDALI *VOICE RECOGNITION*

Melalui lembar ini, penulis ingin menyampaikan ucapan terima kasih dan penghormatan yang sebesar-besarnya kepada:

1. Allah Subhanahu Wa Ta'ala serta junjungan Nabi Muhammad Shollallohu 'Alaihi Wasallam, karena limpahan rahmat dan karunia-Nya penulis dapat menyelesaikan Tugas Akhir dan juga perkuliahan di Departemen Informatika ITS.
2. Ayah dan Ibu penulis, Syawaluddin dan Dedeh yang tiada hentinya memberikan dukungan doa, moral, dan material kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
3. Bapak Dr. Eng. Darlis Herumurti S.Kom., M.Kom. dan Bapak Imam Kuswardayan S.Kom., M.T. selaku pembimbing I dan II yang telah membimbing dan memberikan motivasi, nasihat dan bimbingan dalam menyelesaikan Tugas Akhir ini.
4. Teman teman penulis yang selalu mendorong dan membantu penulis untuk menyelesaikan Tugas Akhir ini.
5. Kakak kakak kelas penulis yang telah memberikan ilmunya untuk penulis.
6. Adik adik kelas penulis yang telah menghibur penulis.
7. Serta semua pihak yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Bagaimanapun juga penulis telah berusaha sebaik-baiknya dalam menyelesaikan tugas akhir ini. Namun, penulis mohon maaf apabila terdapat kekurangan ataupun kesalahkan yang penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan untuk ke depannya.

Surabaya, Juni 2018

Wildan Lutfi Syariati Firdaus Syawal

DAFTAR ISI

LEMBAR PENGESAHAN ..... **vii**
ABSTRAK ..... **ix**
ABSTRACT ..... **xi**
KATA PENGANTAR ..... **xiii**
DAFTAR ISI ..... **xv**
DAFTAR GAMBAR ..... **xvii**
DAFTAR TABEL ..... **xix**
DAFTAR KODE SUMBER ..... **xxi**
1 BAB I PENDAHULUAN **1**
1.1. Latar Belakang *1*
1.2. Rumusan Permasalahan *1*
1.3. Batasan Permasalahan *2*
1.4. Tujuan *3*
1.5. Manfaat *3*
1.6. Metodologi *3*
1.7. Sistematika Penulisan *5*
2 BAB II TINJAUAN PUSTAKA **7**
2.1. Realitas Virtual *7*
2.2. Unity 3D *8*
2.3. Bahasa Pemrograman C# *9*
2.4. First Person Shooter *9*
2.5. Speech Recognition *9*
2.6. Oculus Rift *11*
3 BAB III DESAIN DAN PERANCANGAN SISTEM **13**
3.1. Analisis Sistem *13*
3.1.1. Spesifikasi Kebutuhan Sistem 13
3.1.2. Identifikasi Pengguna 14
3.2. Perancangan Sistem *14*
3.2.1. Deskripsi Umum Sistem 14
3.2.2. Arsitektur Sistem 15
3.3. Perancangan User Interface dan Aset Permainan *16*
3.3.1. *User Interface* 16
3.3.2. Aset Permainan 17
3.4. Perancangan Skenario Simulasi *19*

3.4.1. Alur Simulasi 19
3.4.2. Aturan Permainan 20
3.5. Perancangan Tampilan Antarmuka 21
3.5.1. Tampilan Menu Permainan 21
3.5.2. Tampilan Cara Bermain 22
3.5.3. Tampilan Daftar Sihir 23
3.5.4. Tampilan Permainan 24
3.5.5. Tampilan Akhir Permainan 27
4 BAB IV IMPLEMENTASI SISTEM 29
4.1. Lingkungan Implementasi 29
4.2. Implementasi Permainan 30
4.2.1. Implementasi Menu Permainan 30
4.2.2. Implementasi Cara Bermain 32
4.2.3. Implementasi Daftar Sihir 33
4.2.4. Implementasi Permainan 33
4.2.5. Implementasi Akhir Permainan 63
5 BAB V PENGUJIAN DAN EVALUASI 68
5.1. Lingkungan Pengujian 68
5.2. Pengujian Fungsionalitas 68
5.2.1. Uji Coba Menu Permainan 69
5.2.2. Uji Coba Permainan 70
5.2.3. Uji Coba Akhir Permainan 75
5.2.4. Hasil Uji Coba 76
5.3. Pengujian Pengguna 77
5.3.1. Skenario Pengujian Pengguna 78
5.3.2. Daftar Penguji Permainan 80
5.3.3. Hasil Pengujian Pengguna 80
5.3.4. Kritik dan Saran Pengguna 83
5.4. Evaluasi Pengujian 84
6 BAB VI KESIMPULAN DAN SARAN 86
6.1. Kesimpulan 86
6.2. Saran 86
DAFTAR PUSTAKA 88
LAMPIRAN 90
BIODATA PENULIS 92

DAFTAR GAMBAR

Gambar 3.1 Arsitektur Sistem 16
Gambar 3.2 Panel yang Digunakan 17
Gambar 3.3 Tombol yang Digunakan 17
Gambar 3.4 Environment yang Digunakan 18
Gambar 3.5 Monster yang Digunakan 19
Gambar 3.6 Sihir yang Digunakan 19
Gambar 3.7 Rancangan Tampilan Menu Permainan 22
Gambar 3.8 Rancangan Tampilan Cara Bermain 23
Gambar 3.9 Rancangan Tampilan Daftar Sihir 23
Gambar 3.10 Rancangan Tampilan Dalam Game 24
Gambar 3.11 Rancangan Tampilan Dalam Game, Tangan Kiri Menghadap Kedepan 25
Gambar 3.12 Rancangan Tampilan Dalam Game, Tangan Kiri Menghadap Kebelakang 25
Gambar 3.13 Rancangan Tampilan Dalam Game, Tangan Kanan Tidak Aktif 26
Gambar 3.14 Rancangan Tampilan Dalam Game, Tangan Kanan Aktif 26
Gambar 3.15 Rancangan Tampilan Dalam Game, Permainan Selesai 27
Gambar 4.1 Tampilan Menu Permainan 30
Gambar 4.2 Tampilan Cara Bermain 32
Gambar 4.3 Tampilan Daftar Sihir 33
Gambar 4.4 Tampilan Dalam Game, Tidak Ada Interaksi 34
Gambar 4.5 Tampilan Dalam Game, Interaksi Tangan Kiri Menghadap Kedepan 35
Gambar 4.6 Tampilan Dalam Game, Interaksi Tangan kiri, Menghadap Kebelakan 35
Gambar 4.7 Tampilan Dalam Game, Interaksi Tangan kanan, Sebelum Mengucapkan Sihir 36
Gambar 4.8 Tampilan Dalam Game, Interaksi Tangan Kanan, Setelah Mengucapkan Sihir 36

Gambar 4.9 Panel Win muncul ketika pemain berhasil menyelesaikan game.....66
Gambar 4.10 Panel lose muncul ketika health bar pemain mencapai nol 66

DAFTAR TABEL

Tabel 3.1 Kebutuhan Fungsional Sistem.....13
Tabel 3.2 Kebutuhan Non Fungsional Sistem.....14
Tabel 5.1 Tabel Lingkungan Pengujian Sistem.....68
Tabel 5.2 Hasil Uji Coba Menu Permainan.....69
Tabel 5.3 Hasil Uji Coba Permainan.....71
Tabel 5.4 Hasil Uji Coba Akhir Permainan.....75
Tabel 5.5 Hasil Evaluasi.....77
Tabel 5.6 Rentang Nilai.....78
Tabel 5.7 Format Kuesioner.....79
Tabel 5.8 Daftar Penguji Permainan 80
Tabel 5.9 Hasil Pengujian Pengguna.....81
Tabel 5.10 Hasil Akhir Pengujian Pengguna 82
Tabel 5.11 Kritik dan Saran Pengguna.....83

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

Kode Sumber 4.1 Menu Utama 32
Kode Sumber 4.3 Memeriksa masukan suara pemain dan mengolahnya 39
Kode Sumber 4.4 Script atur boolean pergerakan moving..... 40
Kode Sumber 4.5 Update movement player dari script Oculus Player Controller 42
Kode Sumber 4.6 Script atur boolean skill dapat dikeluarkan atau tidak 43
Kode Sumber 4.7 Script cast sihir setelah boolean terpenuhi 45
Kode Sumber 4.8 Script atur posisi muncul sihir dan arah sihir . 47
Kode Sumber 4.9 Skript kontrol pergerakan dan animasi dari *monster* biasa 50
Kode Sumber 4.10 Skript kontrol pergerakan dan animasi dari *monster* bos 54
Kode Sumber 4.11 Skript kontrol *health bar* & mana bar *monster* dan player 59
Kode Sumber 4.12 Script kontrol ambil dan pakai *item* dari *monster* 62
Kode Sumber 4.13 Script kendali pemunculan panel win dan lose 64
Kode Sumber 4.14 tampilkan panel lose ketika darah player habis 64
Kode Sumber 4.15 Fungsi Win terpanggil ketika pemain berhasil mendapatkan 3 bos *item* 65

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1. Latar Belakang

Permainan digital merupakan salah satu media hiburan yang sedang berkembang pesat, dan bermain permainan digital merupakan salah satu cara untuk mengisi waktu luang. Hingga saat ini ada banyak jenis permainan digital yang sudah ada, dan juga teknologi yang digunakan dalam pembuatan permainan telah berkembang.

Perkembangan teknologi dari waktu ke waktu mengalami kemajuan yang sangat pesat. Seiring dengan perkembangan itu pula, teknologi yang dipakai untuk membuat permainan terus berkembang dan semakin banyak. Teknologi permainan yang sedang berkembang dan menarik perhatian dari para *gamer* adalah permainan berbasis realitas virtual.

Realitas virtual merupakan teknologi yang dapat membuat pengguna berinteraksi dengan suatu lingkungan yang disimulasikan oleh komputer, suatu lingkungan sebenarnya yang ditiru atau benar-benar suatu lingkungan yang hanya ada dalam imajinasi.

Dalam hal ini penulis ingin membuat sebuah permainan yang berjudul Magus. Magus merupakan sebuah permainan realitas virtual dengan *genre survival game* [1], dengan kendali *voice recognition* dan *hand gesture* untuk menggunakan kemampuan yang dimiliki pemain di dalam permainan.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana merancang *gameplay* Magus dengan fitur 3D *First Person Shooter* (FPS) menggunakan konsep Realitas Virtual?
2. Bagaiamana merancang mekanik dari pemain dengan menggunakan kendali *voice recognition* ?
3. Bagaimana implementasi dari rancangan di atas diselesaikan dengan *Game Engine Unity* ?

1.3. Batasan Permasalahan

Batasan masalah pada tugas akhir ini antara lain:

1. Permainan yang dibuat merupakan permainan yang bekerja di *Personal Computer* (PC) yang mumpuni.
2. Permainan yang dibuat merupakan aplikasi realitas virtual yang membutuhkan Oculus Rift untuk realitas virtual dengan kendali *voice recognition*.
3. Masukann bahasa pada *voice recognition* permainan ini adalah bahasa inggris.
4. Lingkungan pengembangan yang digunakan menggunakan aplikasi Unity 3D lisensi gratis dan bahasa pemrograman C#.
5. Kondisi musuh terbatas hanya diam dan menyerang pemain.
6. Obyek musuh dibagi menjadi dua jenis, yaitu musuh yang kuat dan musuh yang lemah.
7. Terdapat musuh kuat yang dianggap *boss* dan musuh lemah yang dianggap *creep*.
8. Pemain memiliki *health bar* dan *mana bar*, yang apabila pemain terkena serangan maka *health bar* akan berkurang dan apabila mengaktifkan sihir maka *mana bar* akan berkurang.
9. Pemain memiliki berbagai macam sihir untuk digunakan.
10. Permainan akan berakhir apabila *health bar* pemain habis atau pemain berhasil mengalahkan seluruh *boss* yang ada.

11. Sepanjang permainan, pemain memiliki kesempatan untuk mendapatkan barang/*item* pembantu untuk menambahkan *health poin* dan/atau *mana poin.*

1.4. Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah untuk membuat permainan realitas virtual yang mengimplementasikan *voice recognition* sebagai pengaktifan sihir dalam *gameplay* permainan Magus.

1.5. Manfaat

Manfaat dari tugas akhir ini yaitu sebagai berikut:

1. Memberikan *gameplay* baru dengan menambahkan fitur *voice recognition* pada permainan realitas virtual.
2. Memberikan pengalaman baru pada pemain permainan realitas virtual.
3. Mengasah kemampuan pengambilan keputusan dan strategi pemain untuk menyelesaikan/memenangkan permainan.
4. Sebagai sarana hiburan untuk para pemain.

1.6. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. **Studi literatur**

   Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan aplikasi yaitu mengenai *Game Engine* Unity3D, *Library* Windows.Speech, Bahasa Pemrograman C#, *Survival Game, First Person Shooter* (FPS), Realitas Virtual, dan *Voice Recognition.*

2. **Analisis dan desain sistem**

   Pada tahap ini dilakukan analisis dan pendefinisian kebutuhan sistem untuk masalah yang dihadapi, terutama

analisis terkait bagaimana skenario pertarungan yang akan diterapkan pada sistem. Selanjutnya, dilakukan perancangan sistem dengan beberapa tahap sebagai berikut:

a. Mempelajari konsep *Voice Recognition* dengan menggunakan *Library* Windows.Speech.
b. Mempelajari dokumentasi dan tutorial Oculus Rift.
c. Mempelajari dokumentasi dan tutorial Unity3D.
d. Perancangan *gameplay* realitas virtual Magus dengan menambahkan fitur *Voice Recognition*.

3. Implementasi sistem

Pada tahap ini akan dilakukan pembangunan sistem. Sistem yang dimaksud disini, yaitu permainan realitas virtual yang dibangun dengan menggunakan tools Unity 3D dan perangkat Oculus Rift dengan OS Windows 10.

4. Pengujian dan evaluasi

Tahap Pengujian dan evaluasi berisi pengujian aplikasi dan evaluasi berdasarkan hasil pengujian. Pada tahap ini akan dilakukan pengujian dari fungsionalitas perangkat lunak, apakah sesuai dengan yang diharapkan serta tidak diharapkan terdapat *bug*. Pengujian akan dilakukan kepada beberapa mahasiswa Departemen Informatika dan beberapa mahasiswa non Departemen Informatika, mereka akan menjadi penguji dan memainkan permainan Magus. Pengujian dilakukan untuk mengukur tingkat ketepatan pengenalan kata/kalimat dari fitur *voice recognition* yang diterapkan.

5. Penyusunan buku Tugas Akhir

Pada tahap ini dilakukan proses dokumentasi dan pembuatan laporan dari seluruh konsep, tinjauan pustaka, metode, implementasi, proses yang telah dilakukan, pengujian, evaluasi dan hasil-hasil yang telah didapatkan selama pengerjaan tugas akhir.

1.7. Sistematika Penulisan

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini:

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, rumusan masalah, tujuan dan manfaat pembuatan tugas akhir, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

Bab II Tinjauan Pustaka

Bab ini menjelaskan beberapa pustaka-pustaka yang dijadikan penunjang dan berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir.

Bab III Desain dan Perancangan Sistem

Bab ini membahas mengenai desain dan perancangan sistem yang akan dibangun.

Bab IV Implementasi Sistem

Bab ini membahas mengenai bagaimana implementasi sistem dari desain yang sudah dirancang.

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian dari metode yang ditawarkan dalam tugas akhir untuk mengetahui kesesuaian metode dengan data yang ada.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang telah dilakukan. Bab ini juga membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan tugas akhir.

Lampiran

Merupakan bab tambahan yang berisi data atau daftar istilah yang penting pada tugas akhir ini.

BAB II
TINJAUAN PUSTAKA

Bab ini membahas pustaka/teori-teori yang menjadi dasar dalam pembuatan tugas akhir.

2.1. Realitas Virtual

Realitas virtual adalah sebuah teknologi yang membuat pengguna dapat berinteraksi dengan lingkungan yang ada dalam dunia maya yang disimulasikan oleh komputer, sehingga pengguna merasa berada di dalam lingkungan tersebut. Teknologi realitas virtual sejatinya telah banyak diterapkan di beberapa sektor industri seperti kedokteran, penerbangan, pendidikan, arsitek, militer, hiburan, dan lain sebagainya. Realitas virtual sangat membantu dalam menyimulasikan sesuatu yang sulit untuk dihadirkan secara langsung dalam dunia nyata. Tentunya ini dapat membuat lebih praktis dan lebih ekonomis.

Kelebihan utama dari realitas virtual adalah pengalaman yang membuat pengguna merasakan sensasi dunia nyata dalam dunia virtual/maya. Bahkan perkembangan teknologi realitas virtual saat ini memungkinkan tidak hanya indra penglihatan dan pendengaran saja yang bisa merasakan sensasi nyata dari realitas virtual, namun juga indra yang lainnya.

Untuk memunculkan sensasi nyata dari realitas virtual diperlukan perangkat pendukung. Paling tidak dibutuhkan sebuah alat seperti Oculus Rift, HTC Vive, dan lain lain yang mendukung realitas virtual untuk bisa merasakan sensasi realitas virtual. Terdapat 4 elemen penting dalam realitas virtual. Adapun 4 elemen itu adalah sebagai berikut:

a. *Virtual world*, sebuah konten yang menciptakan dunia virtual dalam bentuk *screenplay* maupun *script.*
b. *Immersion*, sebuah sensasi yang membawa pengguna teknologi realitas virtual merasakan ada di sebuah

lingkungan nyata yang padahal fiktif. *Immersion* dibagi dalam 3 jenis, yakni:

- *Mental immersion*, membuat mental penggunanya merasa seperti berada di dalam lingkungan nyata.
- *Physical immersion*, membuat fisik penggunanya merasakan suasana di sekitar lingkungan yang diciptakan oleh realitas virtual tersebut.
- *Mentally immersed*, memberikan sensasi kepada penggunanya untuk larut dalam lingkungan yang dihasilkan realitas virtual.

c. *Sensory feedback*, berfungsi untuk menyampaikan informasi dari *virtual world* ke indera penggunanya. Elemen ini mencakup visual (penglihatan), *audio* (pendengaran) dan sentuhan.

d. *Interactivity*, bertugas untuk merespons aksi dari pengguna, sehingga pengguna dapat berinteraksi langsung dalam medan fiktif atau *virtual world*.

Sebuah teknologi dapat dikatakan sebagai realitas virtual jika sudah memenuhi beberapa persyaratan, seperti tampilan gambar/grafis/visualisasi 3D tampak nyata dan sesuai dengan perspektif dari penggunanya, dan mampu mendeteksi semua gerakan dan respons dari pengguna baik itu gerakan kepala atau bola mata pengguna [2].

2.2. Unity 3D

Unity 3D adalah sebuah aplikasi yang digunakan untuk mengembangkan *game* berbasis *multi-platform*. Unity dapat digunakan untuk membuat sebuah *game* yang bisa digunakan pada perangkat komputer, ponsel pintar android, iPhone, PS3, dan bahkan X-BOX.

Unity merupakan sebuah *tool* yang terintegrasi, untuk membuat *game*, arsitektur bangunan dan simulasi. Unity juga bisa digunakan untuk permainan PC dan permainan *online*. Untuk permainan *online* diperlukan sebuah *plugin*, yaitu Unity Web *Player*, sama halnya dengan Flash *Player* pada browser.

Unity tidak dirancang untuk proses desain atau *modelling*, dikarenakan Unity bukanlah sebuah *tools* untuk mendesain, Unity hanyalah sebuah *game engine* 2D atau 3D. Banyak hal yang bisa dilakukan dengan Unity dengan berbagai fitur yang dimilikinya, seperti adanya fitur *audio reverb zone*, *particle effect*, dan *sky box* untuk menambahkan langit. Fitur *scripting* yang disediakan mendukung 3 bahasa pemrograman, JavaScript, C#, dan Boo [3].

2.3. Bahasa Pemrograman C#

C# (dibaca: c sharp) merupakan sebuah bahasa pemrograman berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari insiatif kerangkan .NET *framework*. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur yang terdapat pada bahasa-bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic dan lain-lain dengan beberapa penyederhanaan [4].

2.4. First Person Shooter

Pengembangan *First Person Shooter* (FPS) dimulai pada tahun 1973, dan 1974, Spasim. Setelah itu, lebih banyak judul bermunculan seperti MIDI Maze tahun 1987, kemudian bersatu dengan genre kekerasan membentuk Wolfenstein 3D tahun 1992, akan tetapi lebih kalah populer dengan permainan Doom. Half Life (1998) dan Half Life 2 (2004) meningkatkan narasi dan elemen teka-teki [5].

2.5. Speech Recognition

Pengenalan ucapan atau pengenalan wicara dalam istilah bahasa Inggrisnya, *Speech Recognition* adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan. Teknologi ini memungkinkan suatu perangkat untuk mengenali dan

memahami kata-kata yang diucapkan dengan cara digitalisasi kata dan mencocokkan sinyal digital tersebut dengan suatu pola tertentu yang tersimpan dalam suatu perangkat. Kata-kata yang diucapkan diubah bentuknya menjadi sinyal digital dengan cara mengubah gelombang suara menjadi sekumpulan angka yang kemudian disesuaikan dengan kode-kode tertentu untuk mengidentifikasikan kata-kata tersebut. Hasil dari identifikasi kata yang diucapkan dapat ditampilkan dalam bentuk tulisan atau dapat dibaca oleh perangkat teknologi sebagai sebuah komando untuk melakukan suatu pekerjaan, misalnya penekanan tombol pada telepon genggam yang dilakukan secara otomatis dengan komando suara.

Alat pengenal ucapan, yang sering disebut dengan *speech recognizer*, membutuhkan sampel kata sebenarnya yang diucapkan dari pengguna. Sampel kata akan didigitalisasi, disimpan dalam komputer, dan kemudian digunakan sebagai basis data dalam mencocokkan kata yang diucapkan selanjutnya. Sebagian besar alat pengenal ucapan sifatnya masih tergantung kepada pembicara. Alat ini hanya dapat mengenal kata yang diucapkan dari satu atau dua orang saja dan hanya bisa mengenal kata-kata terpisah, yaitu kata-kata yang dalam penyampaiannya terdapat jeda antar kata. Hanya sebagian kecil dari peralatan yang menggunakan teknologi ini yang sifatnya tidak tergantung pada pembicara. Alat ini sudah dapat mengenal kata yang diucapkan oleh banyak orang dan juga dapat mengenal kata-kata kontinu, atau kata-kata yang dalam penyampaiannya tidak terdapat jeda antar kata.

Pengenalan ucapan dalam perkembangan teknologinya merupakan bagian dari pengenalan suara (proses identifikasi seseorang berdasarkan suaranya). Pengenalan suara sendiri terbagi menjadi dua, yaitu pengenalan pembicara (identifikasi suara berdasarkan orang yang berbicara) dan pengenalan ucapan (identifikasi suara berdasarkan kata yang diucapkan) [6].

Windows.Speech merupakan salah satu *library* yang digunakan untuk membantu pengenalan suara pada sistem operasi Windows 10 [7].

2.6. Oculus Rift

Oculus Rift adalah sistem realitas virtual yang benar-benar menenggelamkan anda ke dalam dunia maya. Rift adalah peranti layar ikat kepala untuk menampilkan realitas virtual yang saat ini dikembangkan oleh Oculus VR [8].

[Halaman ini sengaja dikosongkan]

BAB III
DESAIN DAN PERANCANGAN SISTEM

Bab ini membahas tentang desain dan perancangan Sistem Realitas Virtual untuk Magus. Pembahasan yang dilakukan meliputi analisis sistem, perancangan sistem, skenario simulasi, perancangan *user interface* dan aset permaian, dan perancangan antar muka sistem.

3.1. Analisis Sistem

Sub bab ini akan membahas tentang analisis kebutuhan sistem, meliputi spesifikasi kebutuhan sistem, baik itu kebutuhan fungsional sistem maupun kebutuhan non-fungsional sistem, dan identifikasi pengguna sistem.

3.1.1. Spesifikasi Kebutuhan Sistem

Pada sistem ini terdapat beberapa kebutuhan fungsional dan kebutuhan non-fungsional yang mendukung berjalannya sistem. Kebutuhan fungsional sistem dapat dilihat pada Tabel 3.1, sedangkan kebutuhan non-fungsional sistem dapat dilihat pada Tabel 3.2.

Tabel *3.1 Kebutuhan Fungsional Sistem*

Kode	Deskripsi
F1	Pemain dapat melihat *menu* utama
F2	Pemain dapat memilih pilihan pada *menu* utama
F3	Pemain dapat melihat cara bermain
F4	Pemain dapat melihat daftar sihir
F5	Pemain dapat keluar dari permainan
F6	Pemain dapat memulai permainan
F7	Pemain dapat memulai kembali permainan yang telah selesai
F8	Pemain dapat bergerak maju dan mundur

Kode	**Deskripsi**
F9	Pemain dapat memberikan perintah berupa sihir dalam Bahasa Inggris
F10	Pemain memiliki *health bar* dan mana bar yang dapat berkurang
F11	Karakter pemain dapat mati dalam permainan.
F12	Pemain dapat memenangkan permainan dengan mengumpulkan tiga buah gem dalam game.
F13	*Monster* dapat menyerang pemain
F14	*Monster* dapat berjalan mendekati pemain apabila pemain berada pada *radius* tertentu.
F15	*Boss* dapat menyerang pemain menggunakan sihir
F16	Tiap musuh yang ada akan menjatuhkan *item* secara random

Tabel *3.2 Kebutuhan Non Fungsional Sistem*

Kode	Deskripsi
NF1	Sistem dapat dijalankan di Windows 10
NF2	Sistem memiliki antar muka yang mudah dipahami

3.1.2. Identifikasi Pengguna

Pengguna yang dapat memainkan permainan Magus ini adalah siapa saja (umum). Sehingga, pengguna berhak menggunakan seluruh fungsionalitas yang terdapat pada sistem.

3.2. Perancangan Sistem

Sub bab ini membahas tentang bagaimana sistem ini dirancang, meliputi deskripsi umum sistem, arsitektur sistem.

3.2.1. Deskripsi Umum Sistem

Magus merupakan permainan realitas virtual berbasis *personal computer* bergenre FPS dan *survival game* yang

memanfaatkan Leap Motion dan *Voice Recognition* sebagai kendalinya. Sistem ini merupakan permainan realitas virtual yang menghadirkan atmosfer dan suasana ketika berada di dunia fantasi.

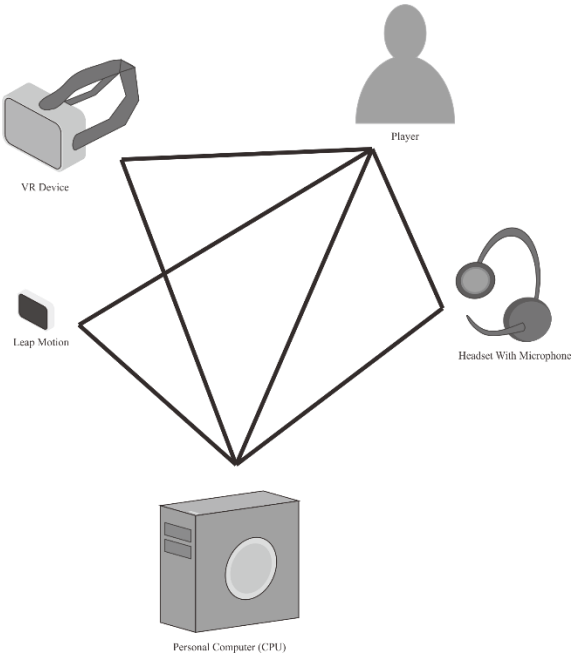
Pembangunan sistem ini dimulai dari membuat *terrain* yang bertema fantasi yang akan dijadikan sebagai lingkungan nyata di dalam permainan VR.

Proses pembangunan sistem selanjutnya yaitu dengan merancang gameplay yang sesuai dengan tema dari permainan ini. Mencari aset aset yang sesuai, dan melengkapi aset dengan membuat secara manual.

3.2.2. Arsitektur Sistem

Magus mengintegrasikan antara *Personal Computer*, Oculus Rift, Leap Motion, dan *Microphone External*. Diproses di PC dan ditampilkan di Oculus Rift, dengan menggunakan Leap Motion, permainan Magus dapat mendeteksi gerakan tangan pemain yang berada depan sensor Leap Motion. Lalu dengan menggunakan *Microphone*, permainan Magus dapat mendengarkan apa yang dikatakan pemain. Dengan begini, permainan Magus lebih terasa nyata, karena menggunakan tangan dan suara sendiri untuk berinteraksi dalam permainan.

Seperti yang dijelaskan diatas, Gambar 3.1 berikut merupakan tampilan diagram dari alur sistem permainan Magus.



Gambar 3.1 Arsitektur Sistem

Dari *player*, sistem menerima input berupa perintah suara melalui *microphone* dan isyarat gerakan tangan melalui Leap Motion, lalu diproses di CPU untuk dijadikan data, hasil dari pemrosesan data tersebut di keluarkan berupa tampilan 3D realitas virtual melalui Oculus Rift dan efek suara melalui *headphone*.

3.3. Perancangan *User Interface* dan Aset Permainan

Pada sub bab ini menjelaskan tentang user interface dan aset yang terdapat dalam permainan. Selain itu akan dibahas pula penggunaannya pada permainan.

3.3.1. *User Interface*

User interface pada permainan ini antara lain:

1. *Panel*

Panel digunakan pada halaman *menu* utama. Gambar 3.2 merupakan tampilan *panel* yang digunakan.

Gambar 3.2 Panel yang Digunakan

2. Tombol

Tombol digunakan pada tiap halaman yang ada dalam permainan. Gambar 3.3 merupakan tampilan dari tombol yang digunakan.

Gambar 3.3 Tombol yang Digunakan

3.3.2. Aset Permainan

Aset pada permainan Magus ini antara lain:

1. *Environment*

Environment digunakan pada tiap halaman permainan yang ada dalam permainan. Gambar 3.4 merupakan tampilan dari environment yang digunakan. *Environment* yang dimaksud ialah *terrain* yang ada pada halaman utama dan dalam permainan.

Gambar 3.4 Environment yang Digunakan

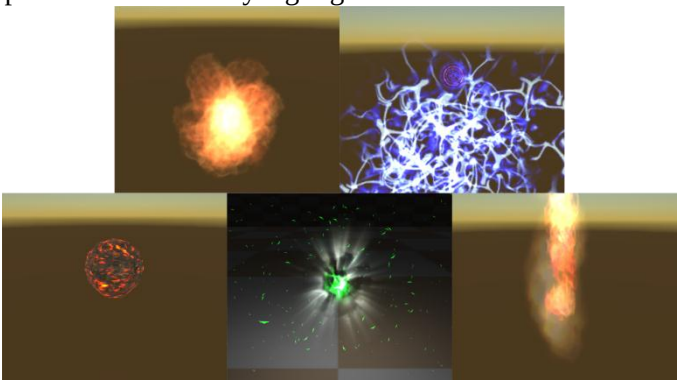
2. *Monster*

Monster dalam permainan Magus ini terdapat banyak macam dan memiliki obyek tersendiri. Gambar 3.5 merupakan tampilan dari *Monster*- *Monster* yang digunakan.

Gambar 3.5 Monster yang Digunakan

3. Sihir

Sihir dalam permainan Magus ini terdapat banyak macam, yaitu *Fire Ball, Lighting Ball, Meteor, Greencore,* dan *Sunstrike* tiap sihir merupakan obyek tersendiri. Gambar 3.6 merupakan tampilan dari sihir sihir yang digunakan.



Gambar 3.6 Sihir yang Digunakan

3.4. Perancangan Skenario Simulasi

Pada sub bab ini menjelaskan tentang skenario permainan untuk menentukan kondisi menang atau kalah. Selain itu akan dibahas pula aturan permainan dari permainan.

3.4.1. Alur Simulasi

Alur permainan dari permainan Magus antara lain:

1. Saat permainan dijalankan maka pemain akan melihat Menu Utama yang memiliki 4 tombol interaksi yaitu, tombol *Play*, *How to Play*, *Skill List*, dan *Exit*.
2. Untuk mengetahui cara bermain pemain menekan tombol *How to Play*. Pemain dapat membaca beberapa petunjuk mengenai cara cara menyelesaikan permainan.

3. Untuk mengetahui daftar sihir yang dapat digunakan, pemain dapat menekan tombol *Skill List*.
4. Untuk bermain pemain menakan tombol *Play*.
5. Setelah memilih tombol main, maka pemain akan menuju halaman utama permainan.
6. Di dalam halaman utama terdapat obyek obyek yang dapat berinteraksi dengan pemain diantaranya adalah *monster*, *item*, serta beberapa tombol.
7. Pada awalnya *monster* akan berada pada posisi *idle*, menunggu interaksi dari pemain.
8. *Monster* akan melakukan sebuah aksi apabila pemain berada didekat *monster*.
9. Ketika *monster* mendeteksi pemain, *monster* akan berjalan menuju pemain lalu menyerang pemain.
10. Pemain diharuskan untuk mengalahkan semua *monster* yang ada dengan menggunakan sihir.
11. Di dalam permainan pemain diharapkan untuk mengumpulkan *item* berupa *gem* yang akan jatuh setelah mengalahkan Bos.
12. Untuk melepaskan sihir, pemain dapat menyebutkan sihir yang ada dengan mengisyaratkan tangan kanan menghadap kedepan.
13. Untuk menggunakan *item heal* dan *mana*, pemain dapat menyebutkan *heal* dan *regen mana*.
14. Pemain dikatakan kalah apabila *health bar* dari pemain telah habis.
15. Pemain dikatakan menang apabila telah berhasil mengumpulkan 3 buah g*em* di dalam permainan.

3.4.2. Aturan Permainan

Dalam memainkan permainan ini, terdapat aturan sebagai berikut:

1. Pada awal permainan, *health* dan *mana* pemain akan berada pada kondisi 100%.

2. Pemain dapat melakukan apa saja, terbatas dengan fitur yang ada.
3. Pemain harus dapat menjaga *health bar* agar tidak habis.
4. Pemain harus mengalahkan semua Bos yang ada dan mendapatkan *item* dari bos tersebut.
5. *Monster* tersebar diseluruh tempat.
6. Jumlah *monster* yang ada tidak diketahui.
7. Besar *health poin monster* tidak diketahui oleh pemain.
8. Untuk melakukan interaksi pemain harus mengisyaratkan tangan di depan Leap Motion dan/atau menyebutkan sihir yang ada.
9. Setiap membunuh *monster*, beberapa barang/*item* penambah *health poin* dan *mana poin* yang kemungkinan muncul ditempat *monster* terbunuh dan pemain dapat mengambil *item* tersebut dengan melewati *item* tersebut.
10. Pemain dapat menggunakan barang penambah *health poin* dan *mana poin* dengan cara sama seperti mengeluarkan sihir.

3.5. Perancangan Tampilan Antarmuka

Sub bab ini membahas bagaimana rancangan antarmuka pengguna yang akan digunakan untuk tugas akhir. Rancangan antarmuka yang dibahas meliputi ketentuan masukan dan rancangan halaman tampilan. Di dalam aplikasi ini terdapat beberapa tampilan, yaitu tampilan menu permainan, tampilan cara bermain, tampilan daftar sihir, tampilan permainan (inti permainan), dan tampilan akhir permainan.

3.5.1. Tampilan Menu Permainan

Tampilan menu permainan merupakan tampilan yang pertama kali muncul ketika aplikasi dijalankan. Pada tampilan awal terdapat empat tombol, yaitu tombol *Play*, *How to Play*, *Skill List*, dan *Exit*. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.2 berikut:

Gambar 3.7 Rancangan Tampilan Menu Permainan

Berikut penjelasan dari Gambar 3.2 :

1. Tombol *Play*, berfungsi untuk memulai permainan.
2. Tombol *How to Play*, berfungsi untuk menampilkan/menyembunyikan tampilan cara bermain di bagian kiri *Play*er.
3. Tombol *Skill List*, berfungsi untuk menampilkan/menyembunyikan tampilan daftar sihir di bagian kanan *Play*er.
4. Tombol *Exit*, berfungsi untuk keluar dari permainan.

3.5.2. Tampilan Cara Bermain

Tampilan cara bermain merupakan halaman yang akan muncul setelah pemain menekan tombol *How to Play*. Halaman ini berisikan informasi cara bermain yang perlu diketahui untuk dapat memainkan permainan. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.3 berikut:

Gambar 3.8 Rancangan Tampilan Cara Bermain

3.5.3. Tampilan Daftar Sihir

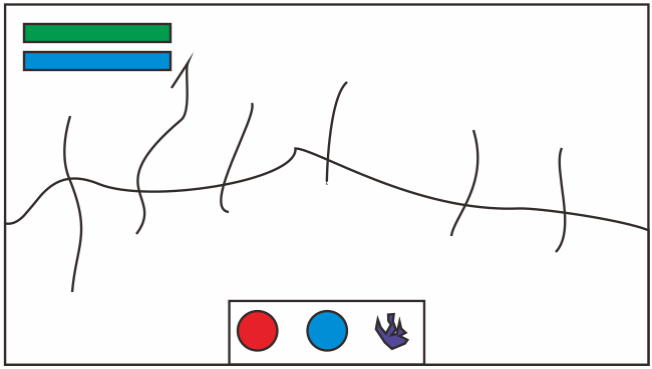
Tampilan daftar sihir merupakan halaman yang akan muncul setelah pemain menekan tombol *Skill List*. Halaman ini berisikan informasi sihir yang perlu diketahui untuk dapat memainkan permainan. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.4 berikut:

Gambar 3.9 Rancangan Tampilan Daftar Sihir

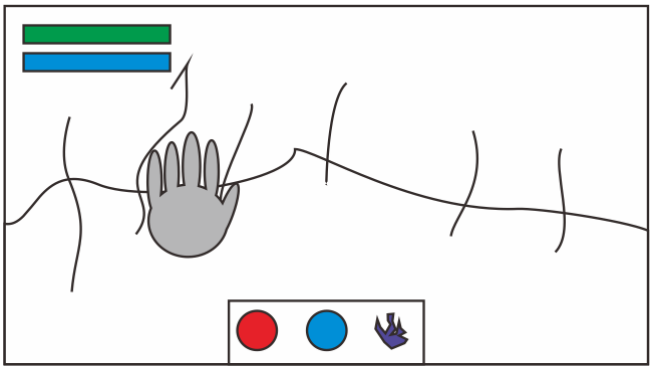
3.5.4. Tampilan Permainan

Tampilan permainan merupakan halaman yang muncul setelah pemain menekan tombol *Play* pada menu permainan. Halaman ini merupakan halaman utama dari permainan, dimana pemain mulai melakukan interaksi terhadap dunia virtual.

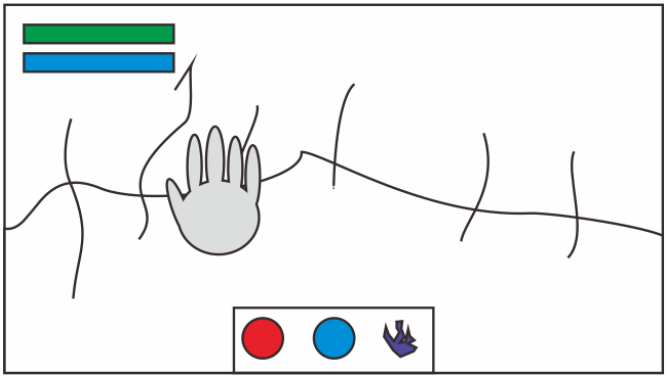
Pada tampilan ini terdapat enam tampilan antarmuka, tampilan rancangan antarmuka dapat dilihat pada Gambar 3.4 – 3.9 berikut:



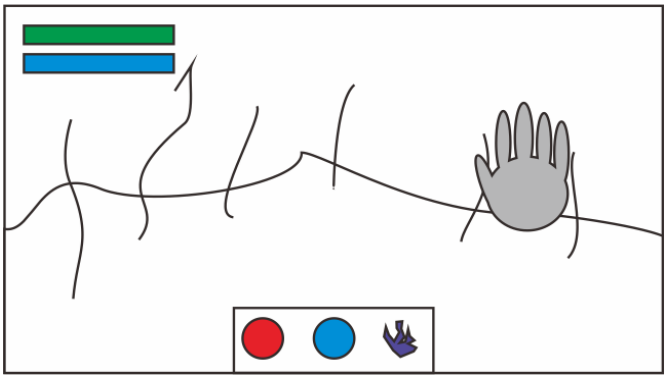
Gambar 3.10 Rancangan Tampilan Dalam Game



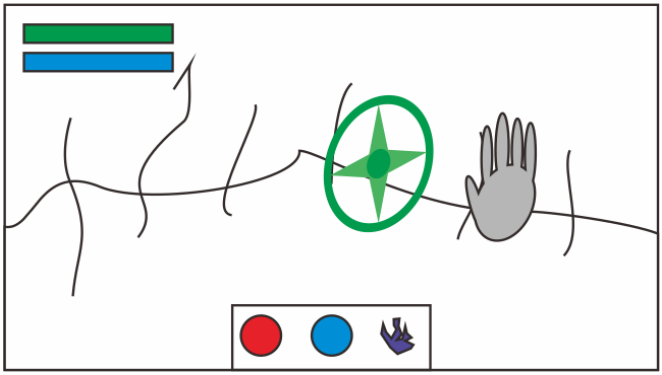
Gambar 3.11 Rancangan Tampilan Dalam Game, Tangan Kiri Menghadap Kedepan



Gambar 3.12 Rancangan Tampilan Dalam Game, Tangan Kiri Menghadap Kebelakang



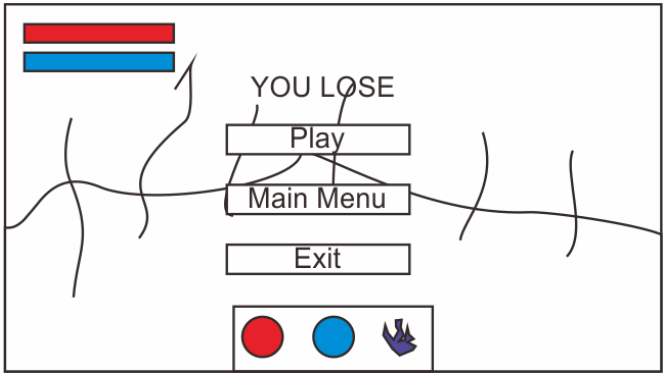
Gambar 3.13 Rancangan Tampilan Dalam Game, Tangan Kanan Tidak Aktif



Gambar 3.14 Rancangan Tampilan Dalam Game, Tangan Kanan Aktif

3.5.5. Tampilan Akhir Permainan

Tampilan akhir permainan merupakan halam ketika pemain mati karena *health bar* habis. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.10 berikut:



Gambar 3.15 Rancangan Tampilan Dalam Game, Permainan Selesai

Berikut merupakat penjelasan tombol tombol yang ada pada Gambar 3.10 :

1. Tombol *Play*, berfungsi untuk mengulangi permainan.
2. Tombol Main Menu, berfungsi untuk kembali ke tampilan menu permainan.
3. Tombol *Exit*, berfungsi untuk keluar dari permainan.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI SISTEM

Bab ini membahas implementasi dari perancangan sistem sesuai dengan perancangan yang telah dibuat. Bahasa pemrograman yang digunakan untuk implementasi sistem adalah Bahasa pemrograman C#.

4.1. Lingkungan Implementasi

Lingkungan implementasi sistem yang digunakan untuk mengembangkan tugas akhir ini memiliki spesifikasi perangkat keras dan perangkat lunak yang ditunjukkan oleh Tabel 4.1.

Tabel 4. 1 Spesifikasi Perangkat

Perangkat	Spesifikasi
Perangkat Keras	• Prosesor: Intel® Core™ i7-7700U CPU @ 3.60GHz (4 CPUs), ~3.6GHz • Memori: 8192MB • VGA : NVIDIA GeForce GTX 1060 3Gb • Oculus Rift DK2 • Leap Motion • *Headphone with Build in Microphone*
Perangkat Lunak	• Sistem Operasi Microsoft Windows 10 64-bit • Perangkat Pengembang Unity3D 2017.3.1 • Perangkat Pembantu Visual Studio Community 2017, Microsoft Word 2017, Corel Draw X7, Adobe After Effects 2017cc, dan Blender

4.2. Implementasi Permainan

Implementasi dari masing-masing fungsi utama dituliskan menggunakan kode berbahasa C#. implementasi fungsi diurut berdasarkan antarmuka-antarmuka yang ada pada permainan.

4.2.1. Implementasi Menu Permainan

Gambar 4.1 merupakan tampilan implementasi dari menu permainan yang akan dijelaskan, sebagai berikut:

Gambar 4.1 Tampilan Menu Permainan

Pada Gambar 4.1 terdapat 4 tombol yaitu, *Play*, *How to Play*, *Skill List*, dan *Exit*. Berikut penjelasan dari tiap tombol:

1. *Play* untuk memulai permainan.
2. *How to Play* untuk memunculkan/menyembunyikan panel cara bermain.
3. *Skill List* untuk memunculkan/menyembunyikan panel daftar sihir.
4. *Exit* untuk keluar dari permainan.

Pada Kode Sumber 4.1, terdapat beberapa fungsi untuk menjalankan halaman awal permainan ini. Diantaranya yaitu untuk pertamacali memulai permaian, fungsi untuk menampilkan panel

cara bermain, fungsi untuk menampilkan panel daftar sihir, dan untuk keluar dari game.

```
1.      public void GoPlay()
2.      {
3.          SceneManager.LoadScene("Main");
4.          Time.timeScale = 1;
5.      }
6.
7.
8.      public void GoMain()
9.      {
10.         SceneManager.LoadScene("Main Menu");
11.         Time.timeScale = 1;
12.     }
13.
14.
15.     public void GoHow2Play()
16.     {
17.         if (flaghow == false)
18.         {
19.             how2Play.SetActive(true);
20.             flaghow = true;
21.         }
22.         else
23.         {
24.             how2Play.SetActive(false);
25.             flaghow = false;
26.         }
27.
28.     }
29.     public void SkillList()
30.     {
31.         if (flagskill == false)
32.         {
33.             skill.SetActive(true);
34.             flagskill = true;
35.         }
36.         else
37.         {
38.             skill.SetActive(false);
39.             flagskill = false;
```

```
        }
    }

    public void Quit()
    {
        Debug.Log("Has quit game");
        Application.Quit();
    }

}
```

Kode Sumber 4.1 Menu Utama

4.2.2. Implementasi Cara Bermain

Tampilan cara bermain yang diimplementasikan dalam permainan dapat dilihat pada Gambar 4.2, berikut:

Gambar 4.2 Tampilan Cara Bermain

Seperti yang ditampilkan pada Gambar 4.2 disampaikan beberapa informasi cara bermain. Untuk kembali ke menu utama pemain cukup melihat kedepan.

4.2.3. Implementasi Daftar Sihir

Tampilan daftar sihir yang diimplementasikan dalam permainan dapat dilihat pada Gambar 4.3, berikut:

Gambar 4.3 Tampilan Daftar Sihir

Seperti yang ditampilkan pada Gambar 4.3 disampaikan beberapa informasi sihir sihir yang dapat digunakan dalam permainan. Untuk kembali ke menu utama pemain cukup melihat kedepan.

4.2.4. Implementasi Permainan

Tampilan permainan merupakan halaman tempat pemain melakukan interaksi. Pada halaman permainan, pemain dapat melakukan berbagai hal seperti jalan dan melepaskan sihir. Tampilan permainan yang diimplementasikan dapat dilihat pada Gambar 4.4 – 4.9.

Gambar 4.4 Tampilan Dalam Game, Tidak Ada Interaksi

Pada Gambar 4.4 terdapat 1 tombol yaitu, Pause.Berikut penjelasan dari tombol:

1. Pause untuk menghentikan permainan sementara

Gambar 4.5 Tampilan setelah pemain menyentuh tombol pause

Pada Gambar 4.5 terdapat 3 tombol yaitu, *Resume*, *Main Menu* dan *Exit*. Berikut penjelasan dari tombol-tombol tersebut:

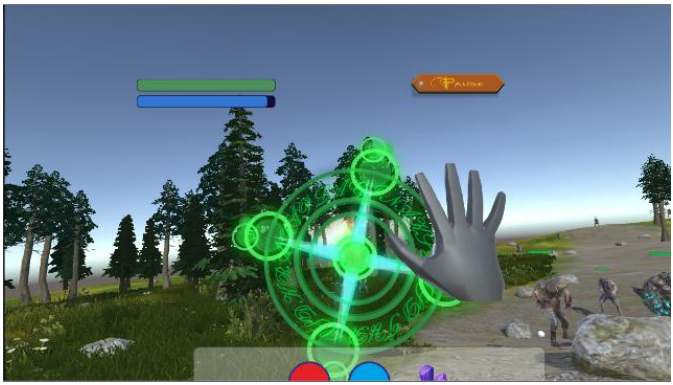
1. *Resume* untuk kembali melanjutkan permainan.
2. *Main Menu* untuk pemain kembali ke *scene Main Menu*.

3. *Exit* untuk keluar dari game.

Gambar 4.5 Tampilan Dalam Game, Interaksi Tangan Kiri Menghadap Kedepan

Gambar 4.6 Tampilan Dalam Game, Interaksi Tangan kiri, Menghadap Kebelakan

Gambar 4.7 Tampilan Dalam Game, Interaksi Tangan kanan, Sebelum Mengucapkan Sihir



Gambar 4.8 Tampilan Dalam Game, Interaksi Tangan Kanan, Setelah Mengucapkan Sihir

Pada Gambar 4.5 – 4.6 terdapat tampilan tangan kiri dari pemain yang mengindikasikan isyarat untuk bergerak. Pada Gambar 4.7 - 4.8 terdapat tampilan tangan kanan dari pemain yang mengindikasikan isyarat untuk melepaskan sihir.

Terdapat beberapa obyek yang ada di dalam permainan, yaitu:

1. Pemain
2. Tangan
3. *Monster*
4. Bos *Monster*
5. *Health bar* pemain
6. *Health bar monster*
7. *Mana bar*
8. Informasi *item*
9. Sihir

Implementasi fungsi dari tiap obyek pada Gambar 4.4 – Gambar 4.8 dapat dilihat di Kode Sumber 4.2 – 4.11 Diantaranya yaitu,

1. Obyek pemain

   Obyek pemain dalam game ini berfungsi sebagai badan virtual dari pemain. Dalam obyek pemain terdapat kode yang dapat membuat system mendengarkan apa kata pemain yang dapat dilihat pada Kode Sumber 4.2.

```
1.  public class PlayerController : MonoBehaviour {
2.
3.  public GameObject[] magicPrefabs = new GameObject
    [5];
4.      public GameObject magicPrefab;
5.      public GameObject magicCircle;
6.      public GameObject tanganKanan;
7.      public AudioSource source;
8.      private float interval = 0.1f;
9.      private FireBaseScript magicPrefabScript;
10.
11.     public Transform magicSpawner;
12.
13.     private string[] keywords = new string[] { "f
    ire ball", "Lighting ball", "meteor", "green core
    ", "sunstrike", "heal", "regen mana" };
14.     public ConfidenceLevel confidence = Confidenc
    eLevel.Low;
15.
16.     private SkillOnActivate scriptCastSkill;
```

```
17.     private PickUp pickUp;
18.     private int mananow, hpnow;
19.     private Health _health;
20.     private bool boolCast;
21.
22.     private float timeRemaining = 0f;
23.
24.     protected PhraseRecognizer recognizer;
25.     protected string word = "";
26.
27.     private void Start()
28.     {
29.         scriptCastSkill = tanganKanan.GetComponen
    t<SkillOnActivate>();
30.         _health = this.gameObject.GetComponent<He
    alth>();
31.         pickUp = this.gameObject.GetComponent<Pic
    kUp>();
32.         if (keywords != null)
33.         {
34.             recognizer = new KeywordRecognizer(ke
    ywords, minimumConfidence: confidence);
35.             recognizer.OnPhraseRecognized += Reco
    gnizer_OnPhraseRecognized;
36.             recognizer.Start();
37.         }
38.     }
39.
40. private void Recognizer_OnPhraseRecognized(Phrase
    RecognizedEventArgs args)
41.     {
42.         word = args.text;
43.     }
44.
45. private void Update()
46.     {
47.         mananow = pickUp.tmana;
48.         hpnow = pickUp.thp;
49.         boolCast = scriptCastSkill.castskill;
50.
51.     private void OnApplicationQuit()
52.     {
```

```
53.         if (recognizer != null && recognizer.IsRu
    nning)
54.         {
55.             recognizer.OnPhraseRecognized -
    = Recognizer_OnPhraseRecognized;
56.             recognizer.Stop();
57.         }
58.     }
59.
60.     void OnDestroy()
61.     {
62.         if (recognizer != null)
63.         {
64.             recognizer.Stop();
65.             recognizer.Dispose();
66.         }
67.     }
68. }
69.
```

Kode Sumber 4.2 Memeriksa masukan suara pemain dan mengolahnya

2. Tangan pemain

Tangan pemain dalam game ini berfungsi sebagai control dalam permainan Magus ini, kedua tangan memiliki fungsi tersendiri,

2.1. Tangan Kiri

Tangan kiri digunakan sebagai kendali pergerakan pemain, jika telapak tangan pemain menghadap ke depan pemain akan bergerak maju dan jika telapak pemain menghadap ke belakang pemain akan bergerak mundur. Implementasi fungsi deteksi tangan kiri apakah dia maju atau mundur dapat dilihat di Kode Sumber 4.3 – Kode Sumber 4.4.

```
1.      public bool movement = false;
2.      public bool backmove = false;
```

```
3.      public void PrintActiveMessage() {
4.          movement = true;
5.          Debug.Log("Maju :" + movement);
6.      }
7.
8.      public void PrintActiveMessageDua()
9.      {
10.         backmove = true;
11.         Debug.Log("Mundur : " + backmove);
12.     }
13.
14.     public void PrintDeactivateMessage() {
15.         movement = false;
16.         Debug.Log("Maju : " +movement);
17.     }
18.
19.     public void PrintDeactivateMessageDua()

20.     {
21.         backmove = false;
22.         Debug.Log("Mundur : " +backmove);
23.     }
24. }
```

Kode Sumber 4.3 Script atur boolean pergerakan moving

```
1.  public virtual void UpdateMovement()
2.  {
3.          boolmoving = scriptPrintJalan.movemen
    t;
4.          boolbackmove = scriptPrintJalan.backm
    ove;
5.
6.          Debug.Log("boolmoving : " + boolmovin
    g);
7.          if (HaltUpdateMovement)
8.           return;
9.
10.     if (EnableLinearMovement)
11.     {
```

```
12.         bool moveForward = Input.GetKey(KeyC
    ode.W) || Input.GetKey(KeyCode.UpArrow);
13.         bool moveLeft = Input.GetKey(KeyCode
    .A) || Input.GetKey(KeyCode.LeftArrow);
14.         bool moveRight = Input.GetKey(KeyCod
    e.D) || Input.GetKey(KeyCode.RightArrow);
15.         bool moveBack = Input.GetKey(KeyCode
    .S) || Input.GetKey(KeyCode.DownArrow);
16.
17.         bool dpad_move = false;
18.
19.         if (OVRInput.Get(OVRInput.Button.Dpa
    dUp))
20.         {
21.             moveForward = true;
22.             dpad_move = true;
23.
24.         }
25.
26.         if (OVRInput.Get(OVRInput.Button.Dpa
    dDown))
27.         {
28.             moveBack = true;
29.             dpad_move = true;
30.         }
31.
32.         MoveScale = 1.0f;
33.
34.         if ((moveForward && moveLeft) || (mo
    veForward && moveRight) ||
35.             (moveBack && moveLeft) || (moveB
    ack && moveRight))
36.             MoveScale = 0.70710678f;
37.
38.         if (!Controller.isGrounded)
39.             MoveScale = 0.0f;
40.
41.         MoveScale *= SimulationRate * Time.d
    eltaTime;
42.         float moveInfluence = Acceleration *
    0.1f * MoveScale * MoveScaleMultiplier;
43.
```

```
44.         if (dpad_move || Input.GetKey(KeyCod
    e.LeftShift) || Input.GetKey(KeyCode.RightSh
    ift))
45.             moveInfluence *= 2.0f;
46.
47.         Quaternion ort = transform.rotation;

48.         Vector3 ortEuler = ort.eulerAngles;

49.         ortEuler.z = ortEuler.x = 0f;
50.         ort = Quaternion.Euler(ortEuler);
51.
52.         if (boolmoving)
53.             MoveThrottle += ort * (transform
    .lossyScale.z * (moveInfluence * 2) * Vector
    3.forward);
54.
55.         if (boolbackmove)
56.             MoveThrottle += ort * (transform
    .lossyScale.z * (moveInfluence * BackAndSide
    Dampen * 2) * Vector3.back);
57.
58.         if (moveLeft)
59.             MoveThrottle += ort * (transform
    .lossyScale.x * moveInfluence * BackAndSideD
    ampen * Vector3.left);
60.         if (moveRight)
61.             MoveThrottle += ort * (transform
    .lossyScale.x * moveInfluence * BackAndSideD
    ampen * Vector3.right);
```

Kode Sumber 4.4 Update movement Player dari script Oculus Player Controller

2.2. Tangan Kanan

Tangan kanan digunakan pemain sebagai kendali terhadap sihir yang akan dikeluarkan pemain untuk menyerang atau lainya. Pada bagian ini penulis menggunakan *library* Windows.Speech, pada *library* ini ada beberapa fungsi yang

digunakan seperti *onPhraseRecognizer*, *keyWordRecognizer*, dan lainnya. Dalam hal ini jika telapak tangan kanan pemain menghadap ke depan, pemain masuk ke kondisi dapat mengeluarkan sihir atau *item*, hanya perlu diikuti kondisi ketersediaan *mana* pemain ataupun *item* pada pemain. Implementasi fungsi untuk mendeteksi apakah pemain dalam kondisi bisa mengeluarkan sihir dari tangan kanan dapat dilihat dari Kode Sumber 4.5 – Kode Sumber 4.7.

```
1.  public class SkillOnActivate : MonoBehaviour
2.  {
3.      public bool castskill = false;
4.
5.
6.      public void PrintActiveMessage()
7.      {
8.          castskill = true;
9.          Debug.Log("Castskill : " +castskill);
10.     }
11.
12.     public void PrintDeactivateMessage()
13.     {
14.
15.         castskill = false;
16.         Debug.Log("Castskill : " +castskill);
17.
18.     }
19. }
```

Kode Sumber 4.5 Script atur boolean skill dapat dikeluarkan atau tidak

```
1.  private void Update()
2.  {
3.      mananow = pickUp.tmana;
4.      hpnow = pickUp.thp;
5.      boolCast = scriptCastSkill.castskill;
6.
7.      if (boolCast && _health.manaOn == true)
```

```
8.      {
9.          switch (word)
10.         {
11.             case "fire ball":
12.                 printCommand(word);
13.                 magicCircle.SetActive(true);
14.                 _health.consume(5);
15.                 cast(0);
16.                 StartCoroutine(Mati());
17.                 break;
18.             case "Lighting ball":
19.                 printCommand(word);
20.                 magicCircle.SetActive(true);
21.                 _health.consume(20);
22.                 cast(1);
23.                 StartCoroutine(Mati());
24.                 break;
25.             case "meteor":
26.                 printCommand(word);
27.                 magicCircle.SetActive(true);
28.                 _health.consume(30);
29.                 updateMeteorSource();
30.                 cast(2);
31.                 StartCoroutine(Mati());
32.                 break;
33.             case "green core":
34.                 printCommand(word);
35.                 magicCircle.SetActive(true);
36.                 selfCast(3);
37.                 _health.heal(50);
38.                 _health.consume(25);
39.                 StartCoroutine(Mati());
40.                 break;
41.             case "sunstrike":
42.                 printCommand(word);
43.                 magicCircle.SetActive(true);
44.                 _health.consume(20);
45.                 cast(4);
46.                 StartCoroutine(Mati());
47.                 break;
48.             case "heal":
49.                 if (hpnow > 0)
50.                 {
```

```
                    printCommand(word);
                    _health.heal(1);
                    pickUp.useItem(1);
                    break;
                }
                else return;
            case "regen mana":
                if (mananow > 0)
                {
                    printCommand(word);
                    _health.heal(2);
                    pickUp.useItem(2);
                    break;
                }
                else return;
        }
    }

    else return;
}
```

Kode Sumber 4.6 Script cast sihir setelah boolean terpenuhi

```
private void cast(int magicIndex) {
    Vector3 pos;
    float yRot = transform.rotation.eulerAngles.y;
    Vector3 forwardY = Quaternion.Euler(0.0f, yRot, 0.0f) * Vector3.forward;
    Vector3 forward = magicSpawner.transform.forward;
    Vector3 right = magicSpawner.transform.right;

    Vector3 up = magicSpawner.transform.up;
    Quaternion rotation = Quaternion.identity;
    magicPrefab = GameObject.Instantiate(magicPrefabs[magicIndex]);
    magicPrefabScript = magicPrefab.GetComponent<FireConstantBaseScript>();
```

```
11.
12.     if (magicPrefabScript == null)
13.     {
14.         magicPrefabScript = magicPrefab.GetCompon
    ent<FireBaseScript>();
15.         if (magicPrefabScript.IsProjectile)
16.         {
17.             rotation = magicSpawner.transform.rot
    ation;
18.             pos = magicSpawner.transform.position
    ;
19.         }
20.         else
21.         {
22.             pos = transform.position + (forwardY
    * 25.0f);
23.         }
24.     }
25.     else
26.     {
27.         pos = transform.position + (forwardY * 10
    .0f);
28.         rotation = transform.rotation;
29.         pos.y = 0.0f;
30.     }
31.
32.     FireProjectileScript projectileScript = magic
    Prefab.GetComponentInChildren<FireProjectileScrip
    t>();
33.     if (projectileScript != null)
34.     {
35.         projectileScript.ProjectileCollisionLayer
    s &= (~UnityEngine.LayerMask.NameToLayer("Friendl
    yLayer"));
36.     }
37.
38.     magicPrefab.transform.position = pos;
39.     magicPrefab.transform.rotation = rotation;
40. }
41.
42. void selfCast(int magicIndex) {
43.     magicPrefab = GameObject.Instantiate(magicPre
    fabs[magicIndex]);
```

```
44.     magicPrefab.transform.position = transform.po
    sition + transform.forward;
45.     magicPrefab.transform.rotation = transform.ro
    tation;
46.     source.Play();
47.     Destroy(magicPrefab, 5.0f);
48. }
```

Kode Sumber 4.7 Script atur posisi muncul sihir dan arah sihir

Pada Kode Sumber 4.8 penulis menggunakan *Library* dari PyroEngine untuk mengimplementasikan sihir sihir yang ada dalam permainan.

3. *Monster*

Monster dalam game ini terdapat dua tipe, *monster* tipe bos dan *monster* tipe biasa, kedua tipe *monster* ini dapat mengetahui keberadaan *Player* dari jarak pandang mereka, mereka pun akan bereaksi jika *Player* menyerang mereka lebih dulu. Implementasi fungsi yang digunakan untuk mengendalikan *monster-monster* dapat dilihat pada Kode Sumber 4.7 - Kode Sumber 4.9.

```
1.  public class AIFix : MonoBehaviour {
2.      public Transform Player;
3.      private Animator animator;
4.      public Transform head;
5.      Rigidbody rb;
6.      public bool attack;
7.      public bool pursuing;
8.
9.      void Start () {
10.         attack = false;
11.         pursuing = false;
12.         if (Player == null)
13.         {
```

```
14.             Player = GameObject.FindGameObjectWit
    hTag("Player").GetComponent<Transform>() as Trans
    form;
15.         }
16.         rb = GetComponent<Rigidbody>();
17.         rb.freezeRotation = true;
18.         animator = GetComponent<Animator>();
19.         rb.isKinematic = true;
20.     }
21.
22.     void Update () {
23.
24.         Vector3 direction = Player.position - thi
    s.transform.position;
25.         direction.y = 0;
26.         float angle = Vector3.Angle(direction, he
    ad.transform.forward);
27.         float temps = Vector3.Distance(Player.pos
    ition, this.transform.position);
28.
29.         if (Vector3.Distance(Player.position, thi
    s.transform.position) < 10 && (angle < 30 || purs
    uing == true ))
30.         {
31.
32.             pursuing = true;
33.             animator.SetBool("isIdle", false);
34.             this.transform.rotation = Quaternion.
    Slerp(this.transform.rotation, Quaternion.LookRot
    ation(direction), 0.1f);
35.
36.
37.             if(direction.magnitude > 2)
38.             {
39.                 rb.isKinematic = false;
40.                 this.transform.position += transf
    orm.forward * 1 * Time.deltaTime;
41.
42.                 animator.SetBool("isWalking", tru
    e);
43.                 animator.SetBool("isAttacking", f
    alse);
44.
```

```
45.             }
46.             else
47.             {
48.                 rb.isKinematic = true;
49.                 animator.SetBool("isAttacking", t
    rue);
50.                 animator.SetBool("isWalking", fal
    se);
51.             }
52.         }
53.         else if(attack == true)
54.         {
55.             pursuing = true;
56.             animator.SetBool("isIdle", false);
57.             this.transform.rotation = Quaternion.
    Slerp(this.transform.rotation, Quaternion.LookRot
    ation(direction), 0.1f);
58.
59.             if (direction.magnitude > 2)
60.             {
61.                 rb.isKinematic = true;
62.                 this.transform.position += transf
    orm.forward * 1 * Time.deltaTime;
63.                 animator.SetBool("isWalking", tru
    e);
64.                 animator.SetBool("isAttacking", f
    alse);
65.             }
66.             else
67.             {
68.                 rb.isKinematic = true;
69.                 this.transform.position = this.tr
    ansform.position;
70.                 animator.SetBool("isAttacking", t
    rue);
71.                 animator.SetBool("isWalking", fal
    se);
72.             }
73.         }
74.         else
75.         {
76.             animator.SetBool("isIdle", true);
```

```
77.             animator.SetBool("isWalking", false);
78.             animator.SetBool("isAttacking", false
    );
79.             pursuing = false;
80.         }
81.     }
82. }
```

Kode Sumber 4.8 Skript kontrol pergerakan dan animasi dari monster biasa

```
1.  public class AIBos : MonoBehaviour {
2.
3.      public Transform Player;
4.      private Animator animator;
5.      public Transform head;
6.      bool pursuing = false;
7.      private bool y = true;
8.      Rigidbody rb;
9.      private bool isCasting = false;
10.
11.     public GameObject[] magic = new GameObject[2]
    ;
12.     public GameObject magicCircle;
13.     private FireBaseScript magicPrefabScript;
14.     private GameObject magicObject;
15.
16.     void Start()
17.     {
18.         rb = GetComponent<Rigidbody>();
19.         rb.freezeRotation = true;
20.         animator = GetComponent<Animator>();
21.         if (Player == null)
22.         {
23.             Player = GameObject.FindGameObjectWit
    hTag("Player").GetComponent<Transform>() as Trans
    form;
24.         }
25.     }
26.
```

```
27.     void Update()
28.     {
29.         Vector3 direction = Player.position - thi
    s.transform.position;
30.         direction.y = 0;
31.         float temps = Vector3.Distance(Player.pos
    ition, this.transform.position);
32.
33.         if (Vector3.Distance(Player.position, thi
    s.transform.position) < 20 || pursuing == true)
34.         {
35.             pursuing = true;
36.             animator.SetBool("isIdle", false);
37.             this.transform.rotation = Quaternion.
    Slerp(this.transform.rotation, Quaternion.LookRot
    ation(direction), 0.1f);
38.             if (direction.magnitude > 2 && direct
    ion.magnitude < 20)
39.             {
40.                 if (!magicCircle.activeSelf) {
41.                     StartCoroutine(processTask())
    ;
42.                 }
43.             }
44.             else
45.             {
46.                 Debug.Log("Masuk ELSE 2");
47.                 animator.SetBool("isIdle", true);

48.                 animator.SetBool("isAttacking2",
    false);
49.             }
50.         }
51.         else
52.         {
53.             animator.SetBool("isIdle", true);
54.             animator.SetBool("isAttacking", false
    );
55.             pursuing = false;
56.         }
57.     }
58.
59.     void flipBool() {
```

```
60.         if (isCasting)
61.         {
62.             isCasting = false;
63.         }
64.         else
65.         {
66.             isCasting = true;
67.         }
68.     }
69.
70.     IEnumerator processTask()
71.     {
72.         int i = 3;
73.
74.         if(y == true)
75.         {
76.             animator.SetBool("isIdle", false);
77.             animator.SetBool("isAttacking2", true
    );
78.             magicCircle.transform.LookAt(Player.p
    osition);
79.             magicCircle.SetActive(true);
80.
81.             StartCoroutine(turnOffMagicCircle());

82.             y = false;
83.         }
84.         else if(y == false)
85.         {
86.             flipBool();
87.             animator.SetBool("isIdle", true);
88.             animator.SetBool("isAttacking2", fals
    e);
89.             yield return new WaitForSeconds(i);

90.             y = true;
91.         }
92.     }
93.
94.     void cast(int magicIndex) {
95.         Vector3 pos;
96.         float yRot = magicCircle.transform.rotati
    on.eulerAngles.y;
```

```
97.         Vector3 forwardY = Quaternion.Euler(0.0f,
    yRot, 0.0f) * Vector3.forward;
98.         Vector3 forward = magicCircle.transform.f
    orward;
99.         Vector3 right = magicCircle.transform.rig
    ht;
100.                Vector3 up = magicCircle.transform
    .up;
101.                Quaternion rotation = Quaternion.i
    dentity;
102.                magicObject = GameObject.Instantia
    te(magic[magicIndex]);
103.                magicPrefabScript = magicObject.Ge
    tComponent<FireConstantBaseScript>();
104.
105.                if (magicPrefabScript == null)
106.                {
107.                    magicPrefabScript = magicObjec
    t.GetComponent<FireBaseScript>();
108.                    if (magicPrefabScript.IsProjec
    tile)
109.                    {
110.                        rotation = magicCircle.tra
    nsform.rotation;
111.                        pos = magicCircle.transfor
    m.position;
112.                    }
113.                    else
114.                    {
115.                        pos = transform.position +
    (forwardY * 25.0f);
116.                    }
117.                }
118.                else
119.                {
120.                    pos = transform.position + (fo
    rwardY * 10.0f);
121.                    rotation = transform.rotation;

122.                    pos.y = 0.0f;
123.                }
124.
```

```
125.                FireProjectileScript projectileScr
    ipt = magicObject.GetComponentInChildren<FirePro
    ectileScript>();
126.                if (projectileScript != null)
127.                {
128.                    projectileScript.ProjectileCol
    lisionLayers &= (~UnityEngine.LayerMask.NameToLay
    er("FriendlyLayer"));
129.                }
130.
131.                magicObject.transform.position = p
    os;
132.                magicObject.transform.rotation = r
    otation;
133.            }
134.
135.
136.            IEnumerator turnOffMagicCircle()
137.            {
138.                yield return new WaitForSeconds(2)
    ;
139.                cast(0);
140.                magicCircle.SetActive(false);
141.            }
142.        }
```

Kode Sumber 4.9 Skript kontrol pergerakan dan animasi dari monster bos

4. *Health bar* dan *mana bar*

Health dan *mana bar* yang ada di game baik pada Pemain dan *Monster* diatur agar setiap *health bar* mencapai nol, *monster* akan mati , atau pemain akan mengalami *game over*. Jika *mana bar* pemain kosong, pemain tidak mungkin dapat mengeluarkan sihir, dan jika pemain menggunakan *item* penambah *health* dan *mana*, *mana* dan *health Play*er akan bertambah mengikuti jumlah *item* yang pemain gunakan. Implementasi fungsi dari *health* dan *mana bar* yang

digunakan pemain dan *monster* dapat dilihat dari Kode Sumber 4.10.

```
1.  public class Health : MonoBehaviour {
2.      private int minHealth = 1;
3.      private int minMana = 1;
4.      private int maxMana = 100;
5.
6.      public GameObject[] item;
7.      public GameObject audiopleyer;
8.      public int maxHealth = 100;
9.      public Slider enemyHealthBar;
10.     public Slider enemyManaBar;
11.     Animator anim;
12.     private AudioPlayer ap;
13.     public bool manaOn = true;
14.     private AIFix ai = new AIFix();
15.     private void Start()
16.     {
17.         if(audiopleyer == null)
18.         {
19.             return;
20.         }
21.         enemyHealthBar.value = maxHealth;
22.         anim = GetComponent<Animator>();
23.         ai = GetComponent<AIFix>();
24.         ap = audiopleyer.GetComponent<AudioPlayer
    >();
25.     }
26.
27.     public void takeDamage(int amount)
28.     {
29.         maxHealth -= amount;
30.         ai.attack = true;
31.         if (maxHealth < minHealth)
32.         {
33.             anim.SetBool("isDead", true);
34.             Destroy(this.gameObject, 3.0f);
35.             Debug.Log(item.Length);
36.             if (item.Length > 0)
37.             {
```

```
38.                 spawnRandom();
39.             }
40.         }
41.         enemyHealthBar.value = maxHealth;
42.     }
43.
44.     public void takeDamageBos(int amount)
45.     {
46.
47.         maxHealth -= amount;
48.         enemyHealthBar.value = maxHealth;
49.         Debug.Log("HP " + enemyHealthBar.value);

50.         if (enemyHealthBar.value <= 0)
51.         {
52.             anim.SetBool("isDead", true);
53.             Destroy(this.gameObject, 3.0f);
54.             spawnItem();
55.         }
56.     }
57.
58.     public void heal(int a)
59.     {
60.         if (a == 1)
61.         {
62.             maxHealth += 25;
63.             enemyHealthBar.value = maxHealth;
64.         }
65.         if( a == 2)
66.         {
67.             maxMana += 25;
68.             enemyManaBar.value = maxMana;
69.         }
70.         if( a == 50)
71.         {
72.             maxHealth += 50;
73.             enemyHealthBar.value = maxHealth;
74.         }
75.     }
76.
77.
78.     public void consume(int cons)
79.     {
```

```
80.         if (maxMana - cons >= 0 && maxMana >= con
    s)
81.         {
82.             maxMana -= cons;
83.             enemyManaBar.value = maxMana;
84.             if (maxMana == 0)
85.             {
86.                 manaOn = false;
87.             }
88.             else manaOn = true;
89.
90.             Debug.Log(enemyManaBar.value + "Caste
    d");
91.
92.         }
93.         else
94.         {
95.             manaOn = false;
96.             Debug.Log(enemyManaBar.value);
97.
98.         }
99.     }
100.
101.            public void takeDamagePlayer(int amoun
    t)
102.            {
103.                maxHealth -= amount;
104.                enemyHealthBar.value = maxHealth;

105.                if (maxHealth < minHealth)
106.                {
107.                    Debug.Log("DEAD");
108.                    ap.PlayLose();
109.                    Time.timeScale = 0;
110.                }
111.            }
112.
113.            void spawnRandom()
114.            {
115.                int totalspawn = Random.Range(0, 5
    );
116.                Debug.Log(totalspawn);
```

```
117.                for (int i = 0; i < totalspawn; i+
    +)
118.                {
119.                    int randomInt = Random.Range(0
    , item.Length);
120.                    Vector3 itemPos = new Vector3(
    this.transform.position.x + Random.Range(-
    1.0f, 1.0f),
121.
    this.transform.position.y + 0.3f, this.transform.
    position.z + Random.Range(-1.0f, 1.0f));
122.                    Instantiate(item[randomInt], i
    temPos, Quaternion.identity);
123.                }
124.            }
125.
126.
127.            void spawnItem()
128.            {
129.
130.                    Vector3 itemPos = new Vector3(
    this.transform.position.x + Random.Range(-
    1.0f, 1.0f),
131.
    this.transform.position.y + 0.3f, this.transform.
    position.z + Random.Range(-1.0f, 1.0f));
132.                    Instantiate(item[0], itemPos,
    Quaternion.identity);
133.            }
134.
135.
136.            public void healingItems(int amount)
137.            {
138.                maxHealth += amount;
139.                enemyHealthBar.value = maxHealth;

140.            }
141.
142.            public void healingmanaItems(int amoun
    t)
143.            {
144.                maxMana += amount;
145.                enemyManaBar.value = maxMana;
```

```
146.            }
147.
148.        }
```

Kode Sumber 4.10 Skript kontrol health bar & mana bar monster dan Player

5. Pick *Item*

Setelah pemain membunuh *monster*, secara *random monster* akan menjatuhkan *item*, berupa *item* penambah darah atau *mana*. Untuk mengambil *item*, pemain cukup melewati *item* tersebut dan penanda jumlah *item* di interface *layer* pemain akan bertambah sejumlah *item* yang didapat. Untuk menggunakan *item* pemain hanya perlu mengarahkan tangan seperti ingin mengeluarkan sihir dan mengatakan "*heal*" untuk menambah *health* atau "*regen mana*" untuk menambah *mana*.

Implementasi fungsi efek mengambil *item* dan penggunaan *item* untuk panel *interface* pemain dapat dilihat di Kode Sumber 4.11.

```
1.  public class PickUp : MonoBehaviour {
2.      public GameObject inventoryPanel;
3.      public GameObject[] inventoryIcons;
4.      public int tmana = 0 , thp = 0, tboss = 0
    ;
5.      private AudioPlayer apx;
6.      public GameObject ap;
7.
8.      private void Start()
9.      {
10.         apx = ap.GetComponent<AudioPlayer>();

11.     }
12.
13.     private void OnTriggerEnter(Collider othe
    r)
14.     {
```

```
15.         foreach (Transform child in inventory
    Panel.transform)
16.         {
17.             if (child.gameObject.tag == other
    .gameObject.tag)
18.             {
19.                 if (child.gameObject.tag == "
    hp")
20.                 {
21.                     string c = child.Find("Te
    xt").GetComponent<Text>().text;
22.                     int tcount = System.Int32
    .Parse(c) + 1;
23.                     thp = tcount;
24.                     child.Find("Text").GetCom
    ponent<Text>().text = "" + tcount;
25.                     Destroy(other.gameObject)
    ;
26.                     return;
27.                 }
28.                 else if(child.gameObject.tag
    == "mana")
29.                 {
30.                     string c = child.Find("Te
    xt").GetComponent<Text>().text;
31.                     int tcount = System.Int32
    .Parse(c) + 1;
32.                     tmana = tcount;
33.                     child.Find("Text").GetCom
    ponent<Text>().text = "" + tcount;
34.                     Destroy(other.gameObject)
    ;
35.                     return;
36.                 }
37.                 else if (child.gameObject.tag
    == "bositem")
38.                 {
39.                     string c = child.Find("Te
    xt").GetComponent<Text>().text;
40.                     int tcount = System.Int32
    .Parse(c) + 1;
41.                     tboss = tcount;
```

```
42.                     child.Find("Text").GetCom
    ponent<Text>().text = "" + tcount;
43.                     Destroy(other.gameObject)
    ;
44.                     if(tcount >= 3)
45.                     {
46.                         apx.PlayWin();
47.                         Time.timeScale = 0;

48.                     }
49.                     return;
50.                 }
51.
52.             }
53.         }
54.     }
55.
56.     public void useItem(int a)
57.     {
58.         if(a == 1)
59.         {
60.
61.             foreach (Transform child in inven
    toryPanel.transform)
62.             {
63.                 if (child.gameObject.tag == "
    hp")
64.                 {
65.                     Debug.Log("HEALTH INCREAS
    E");
66.                     string c = child.Find("Te
    xt").GetComponent<Text>().text;
67.                     int tcount = System.Int32
    .Parse(c) - 1;
68.                     thp = tcount;
69.                     child.Find("Text").GetCom
    ponent<Text>().text = "" + thp;
70.                     return;
71.                 }
72.             }
73.         }
74.         if (a == 2)
75.         {
```

```
76.
77.             foreach (Transform child in inven
    toryPanel.transform)
78.             {
79.                 if (child.gameObject.tag == "
    mana")
80.                 {
81.                     Debug.Log("MANA INCREASE"
    );
82.                     string c = child.Find("Te
    xt").GetComponent<Text>().text;
83.                     int tcount = System.Int32
    .Parse(c) - 1;
84.                     tmana = tcount;
85.                     child.Find("Text").GetCom
    ponent<Text>().text = "" + tmana;
86.                     return;
87.                 }
88.             }
89.         }
90.
91.     }
92. }
```

Kode Sumber 4.11 Script kontrol ambil dan pakai item dari monster

4.2.5. Implementasi Akhir Permainan

Tampilan akhir permainan memiliki dua opsi, yaitu kalah dan menang. Jika pemain berhasil mengalahkan *monster* dan mendapat 3 *item* bos, maka *panel win* akan muncul dan *game* berakhir. Jika pemain kalah atau mencapai *health bar* nol maka panel lose akan muncul dan permainan berakhir. Saat panel muncul pemain dapat memilih ingin bermain lagi atau kembali ke *Main Menu* awal. Implementasi fungsi kendali tampilan *panel win* dan *lose* dapat dilihat di Kode Sumber 4.12 – Kode Sumber 4.14.

```
1.  public class AudioPlayer : MonoBehaviour {
2.
3.      public AudioClip[] MusicClip = new AudioClip[
    3];
4.      public AudioSource MusicSource;
5.      public GameObject PanelLose;
6.      public GameObject PanelWin;
7.      public int musicFlag = 0;
8.      // Use this for initialization
9.      void Start () {
10.         MusicSource.clip = MusicClip[0];
11.         MusicSource.loop = true;
12.         MusicSource.volume = 1;
13.         MusicSource.PlayOnAwake = true;
14.         MusicSource.Play();
15.     }
16.
17.     // Update is called once per frame
18.     public void PlayLose()
19.     {
20.
21.         MusicSource.clip = MusicClip[2];
22.         MusicSource.volume = 1;
23.         MusicSource.loop = false;
24.         MusicSource.Play();
25.         PanelLose.SetActive(true);
26.     }
27.
28.     public void PlayWin()
29.     {
```

```
30.         MusicSource.Stop();
31.         MusicSource.clip = MusicClip[1];
32.         MusicSource.volume = 1;
33.         MusicSource.loop = false;
34.         MusicSource.Play();
35.         PanelWin.SetActive(true);
36.     }
37.
38. }
```

Kode Sumber 4.12 Script kendali pemunculan panel win dan lose

```
1.  public void takeDamagePlayer(int amount)
2.  {
3.      maxHealth -= amount;
4.      enemyHealthBar.value = maxHealth;
5.      if (maxHealth < minHealth)
6.      {
7.          Debug.Log("DEAD");
8.          ap.PlayLose();
9.          Time.timeScale = 0;
10.     }
11. }
```

Kode Sumber 4.13 tampilkan panel lose ketika darah Player habis

```
1.  else if (child.gameObject.tag == "bositem")
2.  {
3.      string c = child.Find("Text").GetComponent<Te
    xt>().text;
4.      int tcount = System.Int32.Parse(c) + 1;
5.      tboss = tcount;
6.      child.Find("Text").GetComponent<Text>().text
    = "" + tcount;
7.      Destroy(other.gameObject);
8.      if(tcount >= 3)
9.      {
10.         apx.PlayWin();
```

```
11.         Time.timeScale = 0;

12.     }
13.     return;
14. }
```

Kode Sumber 4.14 Fungsi Win terpanggil ketika pemain berhasil mendapatkan 3 bos item

Tampilan akhir permainan baik menang ataupun kalah yang diimplementasikan dapat dilihat pada Gambar 4.9 – 4.10.

Gambar 4.9 Panel Win muncul ketika pemain berhasil menyelesaikan game

Gambar 4.10 Panel lose muncul ketika health bar pemain mencapai nol

Pada Gambar 4.9 dan Gambar 4.10 terdapat 3 tombol yaitu, *Play*, *Main Menu*, *Exit*. Berikut penjelasan dari tiap tombol:

1. *Play* untuk memulai permainan baru.

2. *Main Menu* untuk kembali kehalaman awal menu permainan game.
3. *Exit* untuk keluar dari permainan

BAB V
PENGUJIAN DAN EVALUASI

Pada bab ini akan dijelaskan mengenai rangkaian uji coba dan evaluasi yang dilakukan. Proses pengujian dilakukan menggunakan metode *blackbox* berdasarkan skenario yang telah ditentukan.

5.1. Lingkungan Pengujian

Lingkungan penguijan sistem pada pengerjaan tugas ini dilakukan pada lingkungan dan alat kakas pada Tabel 5.1 berikut:

Tabel *5.1 Tabel Lingkungan Pengujian Sistem*

Perangkat	Spesifikasi
Perangkat Keras	• Prosesor: Intel® Core™ i7-7700U CPU @ 3.60GHz (4 CPUs), ~3.6GHz • Memori: 8192MB • VGA : NVIDIA GeForce GTX 1060 3Gb • Oculus Rift DK2 • Leap Motion • Headphone with Build in Microphone
Perangkat Lunak	• Sistem Operasi Microsoft Windows 10 64-bit • Perangkat Pengembang Unity3D 2017.3.1 • Perangkat Pembantu Visual Studio Community 2017, Microsoft Word 2017, Corel Draw X7, Adobe After Effects 2017cc, dan Blender

5.2. Pengujian Fungsionalitas

Pengujian fungsionalitas sistem dilakukan dengan menyiapkan sejumlah skenario sebagai tolok ukur keberhasilan

pengujian. Pengujian fungsionalitas dilakukan dengan mengacu pada sub bab 3.3. Pengujian fungsionalitas yang terdapat pada permainan dijabarkan sebagai berikut.

5.2.1. Uji Coba Menu Permainan

Pada sub bab ini dijelaskan secara detil mengenai skenario yang dilakukan dan hasil yang didapatkan dari pengujian fungsionalitas perangkat lunak yang dibangun pada halaman awal. Penjelasan disajikan dengan menampilkan kondisi awal, masukan, keluaran, hasil yang dicapai, dan kondisi akhir.

Pada menu permainan yang akan diuji adalah fungsionalitas tombol yang terdapat di menu utama, yaitu tombol *Play*, *How to Play*, *Skill List*, dan *Exit*. Tampilan menu permainan dapat dilihat pada Gambar 4.1. Skenario yang telah diuji terdapat pada Tabel 5.2.

Tabel *5.2 Hasil Uji Coba Menu Permainan*

ID	**UF-001**
Nama	Uji Coba Pada Menu Permainan
Tujuan uji coba	Pengguna mengetahui fungsionalitas tombol yang ada pada menu permainan
Kondisi awal	Pemain berada pada menu permainan
Skenario 1	***Pemain memilih tombol Play***
Masukan	Menekan tombol *Play* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah ke halaman permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berada pada halaman permainan
Skenario 2	***Pemain memilih tombol How to Play***
Masukan	Menekan tombol *How to Play* pada dunia virtual

Keluaran yang diharapkan	Muncul panel berupa informasi cara bermain di sebelah kiri pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain dapat membaca panel informasi
Skenario 3	***Pemain memilih tombol Skill List***
Masukan	Menekan tombol *Skill List* pada dunia virtual
Keluaran yang diharapkan	Muncul panel berupa informasi daftar sihir di sebelah kanan pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain dapat membaca panel informasi
Skenario 4	***Pemain memilih tombol Exit***
Masukan	Menekan tombol *Exit* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah keluar dari permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berpindah keluar permainan

Hasil uji dari skenario 1 berpindah ke halaman permainan dapat dilihat pada Gambar 4.4, skenario 2 muncul panel informasi cara bermain dapat dilihat pada Gambar 4.2, skenario 3 muncul panel informasi daftar sihir dapat dilihat pada Gambar 4.3, dan skenario 4 keluar dari permainan, pemain akan keluar dari permainan.

5.2.2. Uji Coba Permainan

Pada sub bab ini dijelaskan mengenai skenario yang dilakukan dan hasil yang didapatkan dari pengujian fungsionalitas pada permainan. Penjelasan disajikan dengan menampilkan

kondisi awal, masukan, keluaran, hasil yang dicapai, dan kondisi akhir. Skenario yang telah diuji terdapat pada Tabel 5.3.

Tabel *5.3 Hasil Uji Coba Permainan*

ID	**UF-002**
Nama	Uji Coba Pada Permainan
Tujuan uji coba	Pengguna mengetahui fungsionalitas interaksi yang ada pada permainan
Kondisi awal	Pemain berada pada halaman permainan
Skenario 1	***Pemain bergerak maju***
Masukan	Memajukan tangan kiri menghadap kedepan
Keluaran yang diharapkan	Pemain bergerak maju sesuai arah pandang
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain bergerak maju sesuai arah pandang
Skenario 2	***Pemain bergerak mundur***
Masukan	Memajukan tangan kiri menghadap kebelakang
Keluaran yang diharapkan	Pemain bergerak mundur sesuai arah pandang
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain bergerak mundur sesuai arah pandang
Skenario 3	***Pemain melepaskan sihir***
Masukan	Memajukan tangan kanan menghadap kedepan lalu mengucapkan kata sihir
Keluaran yang diharapkan	Muncul sihir disekitar pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Muncul sihir disekitar pemain

Skenario 4	***Pemain mengalahkan musuh***
Masukan	Memajukan tangan kanan menghadap ke musuh lalu mengucapkan kata sihir.
Keluaran yang diharapkan	*Health bar* musuh berkurang sesuai dengan nilai kerusakan sihir
Hasil uji coba	Berhasil
Kondisi Akhir	*Health bar* musuh berkurang sesuai dengan nilai kerusakan sihir
Skenario 5	***Pemain dikalahkan oleh musuh***
Masukan	Pemain bergerak menuju musuh
Keluaran yang diharapkan	Musuh melakukan serangan kepada pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Musuh melakukan serangan kepada pemain
Skenario 6	***Musuh menjatuhkan item***
Masukan	Pemain mengalahkan musuh
Keluaran yang diharapkan	Musuh menjatuhkan *item* secara random
Hasil uji coba	Berhasil
Kondisi Akhir	Musuh menjatuhkan *item* secara random
Skenario 7	***Pemain menggunakan item penambah darah***
Masukan	Memajukan tangan kanan menghadap kedepan lalu mengucapkan “Heal”
Keluaran yang diharapkan	*Health bar* pemain bertambah
Hasil uji coba	Berhasil
Kondisi Akhir	*Health bar* pemain bertambah

Skenario 8	***Pemain menggunakan item penambah mana***
Masukan	Memajukan tangan kanan menghadap kedepan lalu mengucapkan “Regen Mana”
Keluaran yang diharapkan	*Mana bar* pemain bertambah
Hasil uji coba	Berhasil
Kondisi Akhir	*Mana bar* pemain bertambah
Skenario 9	***Menampilkan kondisi kalah***
Masukan	*Health bar* pemain menjadi merah seutuhnya
Keluaran yang diharapkan	Muncul panel informasi kekalahan yang berisi tombol *Play*, Main Menu, dan *Exit*
Hasil uji coba	Berhasil
Kondisi Akhir	Muncul panel informasi kekalahan yang berisi tombol *Play*, Main Menu, dan *Exit*
Skenario 10	***Pemain kondisi menang***
Masukan	Pemain berhasil mengumpulkan tiga buah gem
Keluaran yang diharapkan	Muncul panel informasi kemenangan yang berisi tombol *Play*, Main Menu, dan *Exit*
Hasil uji coba	Berhasil
Kondisi Akhir	Muncul panel informasi kekalahan yang berisi tombol *Play*, Main Menu, dan *Exit*
Skenario 11	***Pemain memilih tombol pause***
Masukan	Menekan tombol Pause pada dunia virtual
Keluaran yang diharapkan	Permainan berhenti sementara. UI Pause Menu muncul yang berisi tombol *Resume*, Main menu, *Exit*.
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain dapat menghentikan permainan sementara.

Skenario 12	***Pemain menekan tombol Resume pada UI Pause Menu***
Masukan	Menekan tombol *Resume* pada dunia virtual
Keluaran yang diharapkan	Pemain dapat melanjutkan kembali permainan yang dihentikan. UI Pause Menu hilang.
Hasil uji coba	Berhasil.
Kondisi akhir	Pemain dapat melanjutkan permainan kembali.
Skenario 13	***Pemain menekan tombol Main Menu pada UI Pause Menu***
Masukan	Menekan tombol *Main Menu* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah ke tampilan *Main Menu* di awal permainan.
Hasil uji coba	Berhasil.
Kondisi akhir	Pemain dapat berpindah ke *Main Menu* awal.
Skenario 14	***Pemain menekan tombol Exit pada UI Pause Menu***
Masukan	Menekan tombol *Exit* pada dunia virtual
Keluaran yang diharapkan	Pemain dapat keluar dari permainan
Hasil uji coba	Berhasil.
Kondisi akhir	Pemain keluar dari permainan.

Hasil dari skenario 1 yaitu pemain bergerak maju. Selanjutnya skenario 2 yaitu pemain bergerak mundur. Selanjutnya skenario 3 yaitu pemain memunculkan sihir berupa *Fire Ball*, *Lighting Ball*, *Meteor*, *Green Core*, atau *Sunstrike*. Selanjutnya skenario 4 yaitu *health bar* musuh akan berkurang, jika *health bar* musuh menjadi merah semua maka musuh akan mati. Selanjutnya skenario 5 yaitu *health bar* pemain akan berkurang apabila pemain terkena serangan dari musuh. Selanjutnya skenario 6 yaitu musuh

menjatuhkan *item* secara random, *item* yang terjatuh berupa *item mana* dan darah yang akan digunakan pada skenario 7 dan skenario 8. Selanjutnya skenario 7 yaitu pemain menggunakan *item* penambah darah untuk menambah *health bar*. Selanjutnya skenario 8 yaitu pemain menggunakan *item* penambah mana untuk menambah *mana bar*. Selanjutnya skenario 9 yaitu *health bar* pemain menjadi merah seutuhnya, pada kondisi ini pemain telah kalah dari permainan. Selanjutnya skenario 10 yaitu pemain berhasil mengumpulkan tiga buah gem yang didapatkan dari tiap Bos, dalam kondisi ini pemain memenangkan permainan.

5.2.3. Uji Coba Akhir Permainan

Pada sub bab ini dijelaskan secara detil mengenai skenario yang dilakukan dan hasil yang didapatkan dari pengujian fungsionalitas perangkat lunak yang dibangun pada halaman awal. Penjelasan disajikan dengan menampilkan kondisi awal, masukan, keluaran, hasil yang dicapai, dan kondisi akhir.

Pada menu permainan yang akan diuji adalah fungsionalitas tombol yang terdapat di akhir permainan, yaitu tombol *Play*, *Main Menu*, dan *Exit*. Tampilan menu permainan dapat dilihat pada Gambar 4.9 dan Gambar 4.10. Skenario yang telah diuji terdapat pada Tabel 5.4.

Tabel *5.4 Hasil Uji Coba Akhir Permainan*

ID	**UF-001**
Nama	Uji Coba Pada Akhir Permainan
Tujuan uji coba	Pengguna mengetahui fungsionalitas tombol yang ada pada akhir permainan
Kondisi awal	Pemain berada pada halaman permainan dalam kondisi menang/kalah
Skenario 1	***Pemain memilih tombol Play***
Masukan	Menekan tombol *Play* pada dunia virtual

Keluaran yang diharapkan	Pemain berpindah ke halaman permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berada pada halaman permainan
Skenario 2	***Pemain memilih tombol Main Menu***
Masukan	Menekan tombol *Main Menu* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah ke halaman menu permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berpindah ke halaman menu permainan
Skenario 3	***Pemain memilih tombol Exit***
Masukan	Menekan tombol *Exit* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah keluar dari permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berpindah keluar permainan

Hasil uji dari skenario 1 berpindah ke halaman permainan, setelah memenangkan ataupun kalah dari permainan, pemain dapat mengulangi permainan. Selanjutnya skenario 2 yaitu pemain berpindah ke halaman menu utama. Selanjutnya skenario 4 keluar dari permainan, pemain akan keluar dari permainan.

5.2.4. Hasil Uji Coba

Pada sub bab ini diberikan hasil evaluasi dari pengujian yang dilakukan pada permainan. Hasil evaluasi dapat dilihat pada Tabel 5.5

Tabel *5.5 Hasil Evaluasi*

ID	Deskripsi	Kemungkinan / Skenario	Perilaku Terlaksana
UF-001	Uji Coba Menu Permainan	Skenario 1	Ya
		Skenario 2	Ya
		Skenario 3	Ya
		Skenario 4	Ya
UF-002	Uji Coba Permainan	Skenario 1	Ya
		Skenario 2	Ya
		Skenario 3	Ya
		Skenario 4	Ya
		Skenario 5	Ya
		Skenario 6	Ya
		Skenario 7	Ya
		Skenario 8	Ya
		Skenario 9	Ya
		Skenario 10	Ya
UF-003	Uji Coba Akhir Permainan	Skenario 1	Ya
		Skenario 2	Ya
		Skenario 3	Ya

5.3. Pengujian Pengguna

Pengujian pada permainan yang dibangun tidak hanya dilakukan pada fungsionalitas yang dimiliki, tetapi juga ditujukan kepada pengguna untuk mencoba secara langsung. Pengujian ini berfungsi sebagai pengujian subjektif yang bertujuan untuk mengetahui tingkat keberhasilan permainan yang dibangun dari sisi pengguna. Hal ini dapat dicapai dengan meminta penilaian dan tanggapan dari pengguna terhadap sejumlah aspek permainan yang ada.

5.3.1. Skenario Pengujian Pengguna

Dalam melakukan pengujian permainan, pengguna diminta mencoba memainkan permainan untuk mencoba semua fungsionalitas dan fitu yang ada. Pengujian permainan oleh pengguna dilakukan dengan sebelumnya memberikan informasi seputar permainan, kegunaan, dan fitur-fitur yang dimiliki. Setelah informasi tersampaikan, pengguna kemudian diarahkan untuk langsung mencoba permainan dengan spesifikasi lingkungan yang sama dengan yang telah diuraikan pada Tabel 5.1 Lingkungan Uji Coba.

Jumlah pengguna yang terlibat dalam pengujian perangkat sebanyak 10 orang. Dalam melakukan pengujian pengguna melakukan percobaan lebih dari satu kali penggunaan untuk masing-masing pengguna.

Dalam memberikan penilaian dan tanggapan, pengguna diberikan kuesioner pengujian permainan. Kuesioner pengujian ini dilakukan secara online melalui *website* google *form* dengan tautan https://intip.in/KuisionerMagus dan hasilnya akan ditampilkan pada tautan https://intip.in/TanggapanKuisionerMagus . Kuesioner pengujian ini memiliki beberapa aspek penilaian seputar desain antarmuka, *immersivity*, dan tingkat kenyamanan permainan. Nilai yang diberikan rentang nilai 1 hingga 6 dengan rincian pada Tabel 5.6 . pada bagian akhir terdapat saran untuk perbaikan fitur. Detil kuesioner pengguna dapat dilihat pada Tabel 5.7.

Tabel *5.6 Rentang Nilai*

No	Keterangan	Nilai
1	Sangat Tidak Setuju (STS)	1
2	Tidak Setuju (TS)	2
3	Kurang Setuju (KS)	3
4	Cukup Setuju (CS)	4
5	Setuju (S)	5
6	Sangat Setuju (SS)	6

Tabel *5.7 Format Kuesioner*

No	Parameter	STS	TS	KS	CS	S	SS
1	Permainan memiliki objek dan latar belakang yang sesuai	0	0	2	1	5	2
2	Permainan memiliki tampilan, warna, dan desain yang menarik	0	0	2	3	4	1
3	Permainan memiliki antarmuka yang mudah dilihat/dikenali	0	0	3	2	4	1
4	Saya merasa seperti penyihir	0	0	0	1	5	4
5	Permainan dapat menangkap suara saya dengan baik						
6	Saya merasakan sensasi menjadi penyihir	0	0	0	1	7	2
7	Permainan dapat berjalan lancar tanpa adanya lag dan atau crash	0	0	4	3	1	2
8	Saya merasa terbantu dengan adanya petunjuk yang ada	0	1	1	3	4	1
9	Saya merasa nyaman selama memainkan permainan ini	0	0	2	3	5	0
10	Saya merasa tertarik untuk memainkan permainan ini untuk selanjutnya	0	0	1	1	5	3

5.3.2. Daftar Penguji Permainan

Pada sub bab ini ditunjukkan daftar pengguna yang bertindak sebagai penguji coba permainan yang dibangun. Dalam pengujian ini tidak terdapat kriteria atau keahlian khusus yang harus dimiliki pengguna karena permainan ini ditunjukkan kepada berbagai kalangan pengguna baik yang suka bermain permainan ataupun tidak. Daftar nama penguji permainan ini dapat dilihat pada Tabel 5.8 .

Tabel *5.8 Daftar Penguji Permainan*

No	Nama	Pekerjaan
1	Muhammad Alfi Maulana Fikri	Mahasiswa
2	Muhammad Roychan Meliaz	Mahasiswa
3	Fintanto Cendekia	Mahasiswa
4	Panji Rimawan	Mahasiswa
5	Aditya Gunawan	Mahasiswa
6	Romi Yehezkiel Purba	Mahasiswa
7	Rahmat Rijal	Mahasiswa
8	Aufar	Mahasiswa
9	Deni Ismail	Mahasiswa
10	Wira Mahardika	Mahasiswa

5.3.3. Hasil Pengujian Pengguna

Sistem penilaian didasarkan pada skala perhitungan satu sampai enam dimana skala satu menunjukkan nilai terendah dan skala enam menunjukkan skala tertinggi. Penilaian akhir kemudian dilakukan dengan menghitung berapa banyak penguji yang memilih suatu skala tertentu dan kemudian dicari nilai rata-ratanya. Hasil uji coba dipaparkan secara lengkap dengan disertai Tabel yang dapat dilihat pada Tabel 5.9 dan Tabel 5.10 .

Tabel *5.9 Hasil Pengujian Pengguna*

No	Pernyataan	Penilaian						Rata-Rata
		1	2	3	4	5	6	
1	Permainan memiliki objek dan latar belakang yang sesuai	0	0	0	2	7	1	4.9
2	Permainan memiliki tampilan, warna, dan desain yang menarik	0	0	1	4	4	1	4.5
3	Permainan memiliki antarmuka yang mudah dilihat/dikenali	0	0	1	5	3	1	4.4
4	Saya merasa seperti penyihir	0	0	0	1	7	2	5.1
5	Permainan dapat menangkap suara saya dengan baik	0	0	1	0	9	0	4.8
6	Saya merasakan sensasi menjadi penyihir	0	0	0	1	8	1	5.0
7	Permainan dapat berjalan lancar tanpa adanya lag dan atau crash	0	0	2	2	6	0	4.4
8	Saya merasa terbantu dengan adanya petunjuk yang ada	0	0	0	6	2	2	4.6
9	Saya merasa nyaman selama memainkan permainan ini	0	0	1	1	8	0	4.7
10	Saya merasa tertarik untuk memainkan	0	0	0	0	7	3	5.3

	permainan ini untuk selanjutnya							

Tabel *5.10 Hasil Akhir Pengujian Pengguna*

No	Pernyataan	Rata-Rata	Total	Total (%)
Parameter Antarmuka				
1	Permainan memiliki objek dan latar belakang yang sesuai	4.9	4.6	76.6%
2	Permainan memiliki tampilan, warna, dan desain yang menarik	4.5		
3	Permainan memiliki antarmuka yang mudah dilihat/dikenali	4.4		
Parameter Immersity				
4	Saya merasa seperti penyihir	5.1	4.9	82.7%
5	Permainan dapat menangkap suara saya dengan baik	4.8		
6	Saya merasakan sensasi menjadi penyihir	5.0		
Parameter Kenyamanan				
7	Permainan dapat berjalan lancar tanpa adanya lag dan atau crash	4.4	4.75	79.1%
8	Saya merasa terbantu dengan adanya petunjuk yang ada	4.6		
9	Saya merasa nyaman selama memainkan permainan ini	4.7		
10	Saya merasa tertarik untuk memainkan permainan ini untuk selanjutnya	5.3		

5.3.4. Kritik dan Saran Pengguna

Dalam memberikan penilain dan tanggapan, pengguna diberikan kuesioner pengujian permainan. Kuesioner pengujian permainan ini terdapat bagian kritik dan saran untuk perbaikan fitur kedepannya. Kritik dan sara penggunan dapat dilihat pada Tabel 5.11 .

Tabel *5.11 Kritik dan Saran Pengguna*

No	Nama	Kritik dan Saran
1	Muhammad Alfi Maulana Fikri	Posisi tangan kurang pas dengan penglihatan sisanya udah bagus, tinggal poles aja
2	Muhammad Roychan Meliaz	Kalo bisa si perkenal fitur satu satu ga langsung bisa dipake semua. Tinggal polish
3	Fintanto Cendekia	Memberikan indikator pada saat *How to Play* dan *Skill List* diklik. Beberapa pohon bisa d*item*bus.
4	Panji Rimawan	Overall sudah bagus dan keren, meskipun masih ada bug-bug kecil di beberapa kondisi. Audio saat bermain kurang memberikan kesan seperti yang ada di permainan. Seharusnya diberikan audio / backsound yang menegangkan seperti yang ada di permainan. UI status *item* kurang terlihat, lebih baik ditempatkan agak ke atas agar terlihat. Begitu juga

		dengan loading icon ketika berganti dari permainan ke main menu.
5	Aditya gunawan	Metode maju atau mundur melelahkan pemain seharusnya ditambahkan maju atau mundur otomatis setelah maju beberapa detik.
6	Romi Yehezkiel Purba	Cara menangkap spelling lebih luas agar casting lebih mudah
7	Rahmat Rijal	aplikasi memiliki design enviroment yang menarik, dan cara menggunakan sihir juga mudah. tetapi kalau bisa untuk berhenti berjalan, sebaiknya menggunakan isyarat menggemgang tangan saja
8	Aufar	sudah mantab kok, keren cuy
9	Deni Ismail	Sensor Leap terkadang kurang presisi
10	Wira Mahardika	Pergerakan pemainya agak sulit

5.4. Evaluasi Pengujian

Sub bab ini membahas mengenai evaluasi terhadap pengujian-pengujian yang telah dilakukan. Dalam hal ini, evaluasi menunjukkan data rekapitulasi dari hasil pengujian fungsionalitas. Rekapitulasi disusun dalam bentuk Tabel yang dapat dilihat pada Tabel 5.5. Dari data yang terdapat pada Tabel tersebut, diketahui bahwa aplikasi yang dibuat telah berjalan sesuai dengan skenario yang diharapkan.

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan tugas akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang

6.1. Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian yang dilakukan, dapat diambil kesimpulan sebagai berikut.

1. Permainan dapat menampilkan tampilan FPS dalam lingkungan realitas virtual.
2. Permainan dapat mendengar suara pemain dengan menggunakan *Voice Recognition* saat kondisi pendeteksian terpenuhi.
3. Permainan dapat melaksanakan perintah sihir dengan baik.
4. Permainan dapat mendeteksi gerakan tangan untuk berinteraksi dengan obyek dalam permainan.
5. Permainan dapat memberikan *feedback* berupa animasi.
6. Didapatkan bahwa pemain dapat merasakan bagaimana menjadi penyihir dengan melepaskan sihir menggunakan tangan dan suara sendiri.
7. Permainan berhasil dibuat dengan *Game Engine Unity.*

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi, dan pengujian yang telah dilakukan.

1. Menambahkan lebih banyak lagi sihir sihir agar menjadi lebih seru.
2. Menstabilkkan gerakan tangan serta obyek pada permainan agar terlihat lebih nyata.
3. Mengoptimalkan *Frame Per Second* (FPS) dari permainan Magus ini, dikarenakan penyusunan kode yang digunakan dalam pengembangan permainan ini kurang bagus, terdapat dampak yang signifikan yaitu turunnya FPS dari permainan.
4. Menambahkan lebih banyak *sound effect* agar permainan terasa lebih nyata.

DAFTAR PUSTAKA

[1] Wikipedia, "Survival Game," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Survival_game. [Diakses 9 Desember 2017].

[2] K. G. D. Herlangga, "Virtual Reality dan Perkembangannya - CodePolitan.com," CodePolitan, 7 Maret 2016. [Online]. Available: https://www.codepolitan.com/virtual-reality-dan-perkembangannya. [Diakses 9 Desember 2017].

[3] Unity, "Unity," Unity, [Online]. Available: https://unity3d.com/unity. [Diakses Desember 2017].

[4] Wikipedia, "Wikipedia C Sharp," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/C_Sharp_(programming_language). [Diakses 9 Desember 2017].

[5] Wikipedia, "Wikipedia First Person Shooter," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/First-person_shooter. [Diakses 9 Desember 2017].

[6] Wikipedia, "Speech Recognition," Wikipedia, [Online]. Available: https://id.wikipedia.org/wiki/Pengenalan_ucapan. [Diakses 18 Januari 2018].

[7] Unity, "Windows.Speech," Unity, [Online]. Available: https://docs.unity3d.com/ScriptReference/Windows.Speech.PhraseRecognizer.Start.html. [Diakses 18 Januari 2018].

[8] Oculus, "Oculus Rift," Oculus, [Online]. Available: https://www.oculus.com/rift/. [Diakses 9 Desember 2017]. [Diakses 9 Desember 2017].

[Halaman ini sengaja dikosongkan]

LAMPIRAN

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Wildan Lutfi Syariati Firdaus Syawal, lahir di Gowa pada tanggal 22 Juni 1996. Lulus dari SMAN 05 Gowa pada tahun 2014 dan melanjutkan studinya di Departemen Informatika Institut Teknologi Sepuluh Nopember Surabaya.

Pernah mendapat gelar Finalis pada MAGE 2017 bidang Pengembangan Game. Aktif mengikuti organisasi antara lain staf Departemen Minat dan Bakat Himpunan Mahasiswa Teknik Computer-Informatik (HMTC) 2015/2016, Staf Departemen Produksi Unit Kegiatan Tari dan Karawitan (UKTK) 2015/2016, dan Staf Dewan Perwakilan Angkatan Himpunan Mahasiswa Teknik Computer-Informatik (DPA-HMTC) 2016-2017.

Dalam menyelesaikan Pendidikan sarjana, penulis mengambil bidang minat Interaksi, Grafika dan Seni (IGS). Penulis dapat dihubungi melalui alamat *e-mail*: wildanlsfs@gmail.com.